Towards the OSA and beyond
Using Wireshark for z/OS Packet Trace Analysis

Matthias Burkhard
mburkhar@de.ibm.com
IBM Germany

Twitter : mreede

August 8. 2012 3:00 PM – 4:00 PM
11341 Platinum Ballroom Salon 9

IP Wizards
ip.wizards@groups.facebook.com

Session Contents

The days are over when connectivity problems in the System z could be solved by z/OS personnel only.
In today's modern multi-tier multi-platform application designs a new approach in network diagnosis is required.

While the z/OS packet trace is always a good start on the quest to the real root cause of a problem, unfortunately outside the zSeries the SYSTCPDA packet trace is not known well enough to serve as a trusted evidence.
This session will demonstrate how the use of wireshark helped to speed up problem resolution for problems that surfaced on z/OS but had their root cause outside the mainframe.

This session is a preparation for the wireshark hands-on lab session today:
10342: Taming the (wire)shark – Orange County Salon 2 at 4:30PM

Some background information – BDP

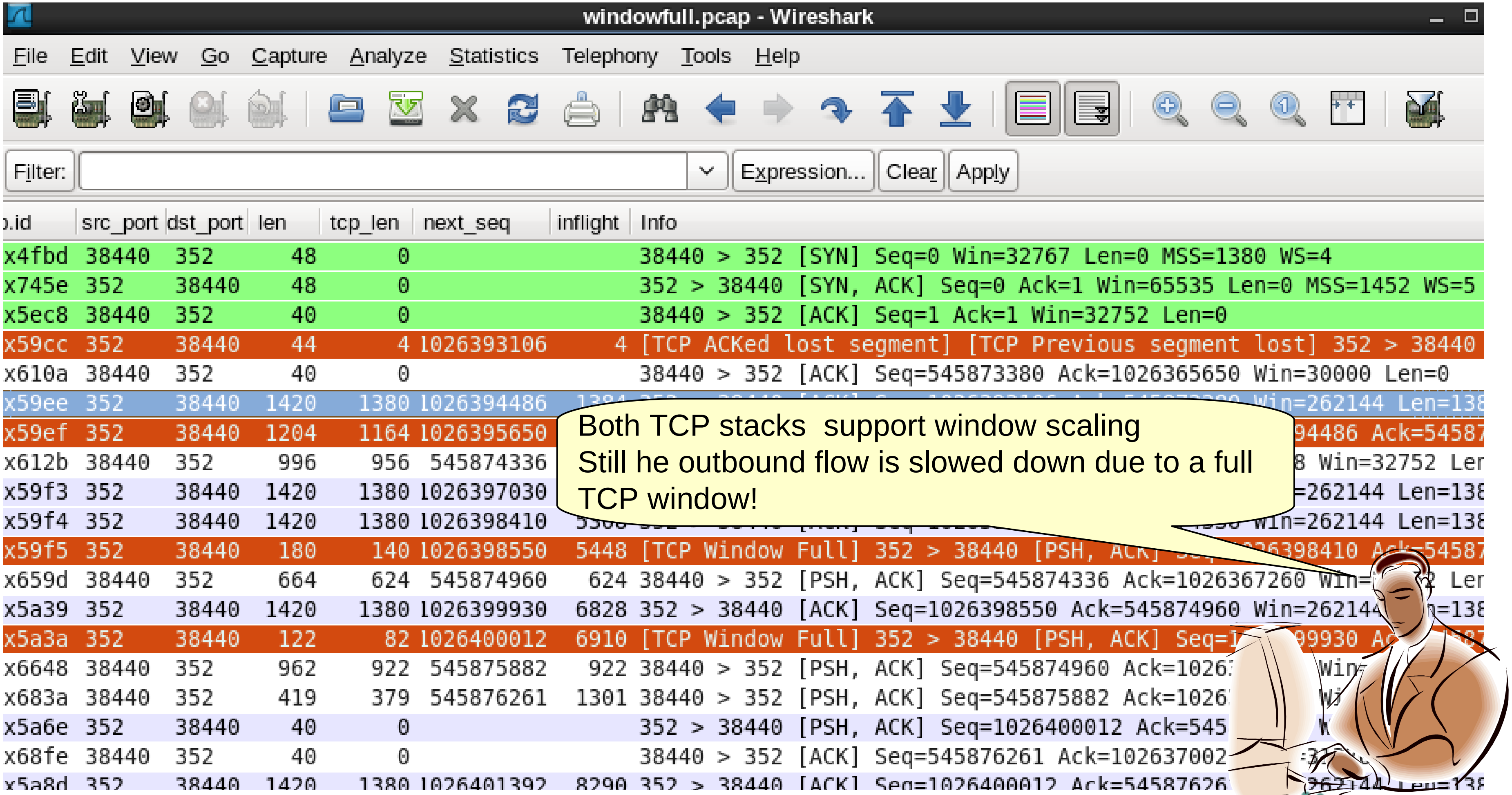
http://en.wikipedia.org/wiki/Bandwidth-delay_product

BDP Bandwidth Delay Product
Available Bandwidth * Network Delay = size of TCP Receivebuffers
Example: 10 Mb/s link with a delay of 0.054 secs requires 70KB buffer for a steady TCP flow, for faster links even more...

A high bandwidth-delay product is an important formula ... because the protocol can only achieve optimum throughput if a sender sends a sufficiently large quantity of data before being required to stop and wait until a confirming message is received from the receiver, acknowledging successful receipt of that data.
If the quantity of data sent is insufficient compared with the bandwidth-delay product, then the link is not being kept busy and the protocol is operating below peak efficiency for the link.

Warmup: TCP Performance - BDP

What is configured? Is it problematic?


windowfull.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply

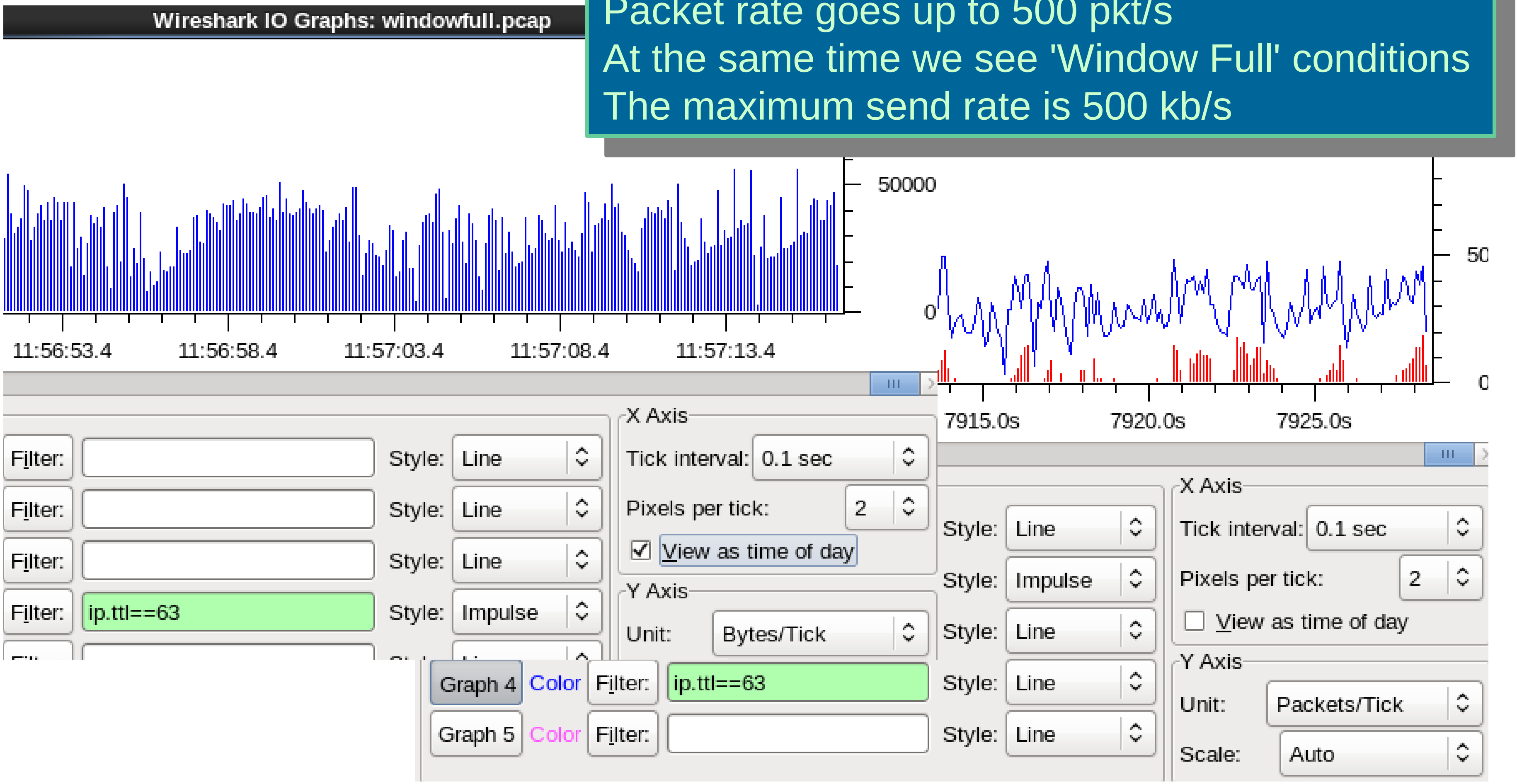
```
.id    src_port dst_port len  tcp_len next_seq   inflight Info
x4fbd  38440    352      48       0                       38440 > 352 [SYN] Seq=0 Win=32767 Len=0 MSS=1380 WS=4
x745e  352      38440    48       0                       352 > 38440 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1452 WS=5
x5ec8  38440    352      40       0                       38440 > 352 [ACK] Seq=1 Ack=1 Win=32752 Len=0
x59cc  352      38440    44       4 1026393106        4 [TCP ACKed lost segment] [TCP Previous segment lost] 352 > 38440
x610a  38440    352      40       0                       38440 > 352 [ACK] Seq=545873380 Ack=1026365650 Win=30000 Len=0
x59ee  352      38440  1420    1380 1026394486     1384 352 > 38440 [ACK] Seq=1026393106 Ack=545873380 Win=262144 Len=138
x59ef  352      38440  1204    1164 1026395650                                                     94486 Ack=54587
x612b  38440    352     996     956  545874336                                                     8 Win=32752 Len
x59f3  352      38440  1420    1380 1026397030                                                     =262144 Len=138
x59f4  352      38440  1420    1380 1026398410                                                     Win=262144 Len=138
x59f5  352      38440   180     140 1026398550     5448 [TCP Window Full] 352 > 38440 [PSH, ACK] Seq=1026398410 Ack=54587
x659d  38440    352     664     624  545874960      624 38440 > 352 [PSH, ACK] Seq=545874336 Ack=1026367260 Win=    2 Len
x5a39  352      38440  1420    1380 1026399930     6828 352 > 38440 [ACK] Seq=1026398550 Ack=545874960 Win=262144  n=138
x5a3a  352      38440   122      82 1026400012     6910 [TCP Window Full] 352 > 38440 [PSH, ACK] Seq=1   99930 Ac
x6648  38440    352     962     922  545875882      922 38440 > 352 [PSH, ACK] Seq=545874960 Ack=1026        Win
x683a  38440    352     419     379  545876261     1301 38440 > 352 [PSH, ACK] Seq=545875882 Ack=1026        Wi
x5a6e  352      38440    40       0                       352 > 38440 [PSH, ACK] Seq=1026400012 Ack=545       W
x68fe  38440    352      40       0                       38440 > 352 [ACK] Seq=545876261 Ack=102637002
x5a8d  352      38440  1420    1380 1026401392     8290 352 > 38440 [ACK] Seq=1026400012 Ack=54587626      262144 Len=138
```

Both TCP stacks support window scaling
Still he outbound flow is slowed down due to a full TCP window!

Warmup: IO Graph

What is the performance?

Packet rate goes up to 500 pkt/s
At the same time we see 'Window Full' conditions
The maximum send rate is 500 kb/s



Filter: Style: Line
Filter: Style: Line
Filter: Style: Line
Filter: ip.ttl==63 Style: Impulse

X Axis
Tick interval: 0.1 sec
Pixels per tick: 2
☑ View as time of day

Y Axis
Unit: Bytes/Tick

Style: Line
Style: Impulse
Style: Line
Graph 4 Color Filter: ip.ttl==63 Style: Line
Graph 5 Color Filter: Style: Line

X Axis
Tick interval: 0.1 sec
Pixels per tick: 2
☐ View as time of day

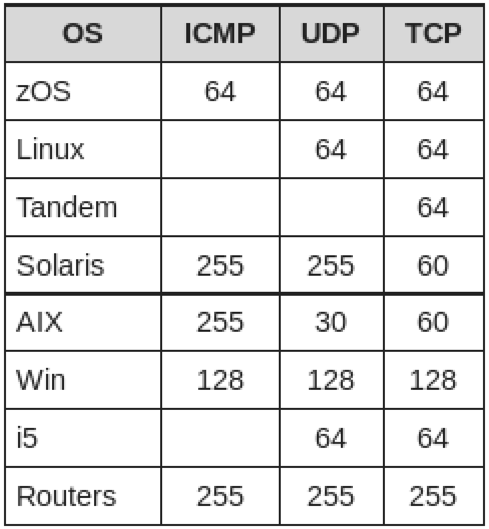
Y Axis
Unit: Packets/Tick
Scale: Auto

Some background information – TTL

http://en.wikipedia.org/wiki/Time_to_live

The Time To Live in the IP header is used to avoid endlessly looping IP datagrams. As a packet gets routed through the network, every Layer 3 device has to decrement the TTL before sending the packet to the next hop.
When a datagram arrives with a TTL of 1 and has not yet reached its final destination, the receiving ip node must discard the datagram and send an ICMP message back to its originator to inform him that the packet is dropped.

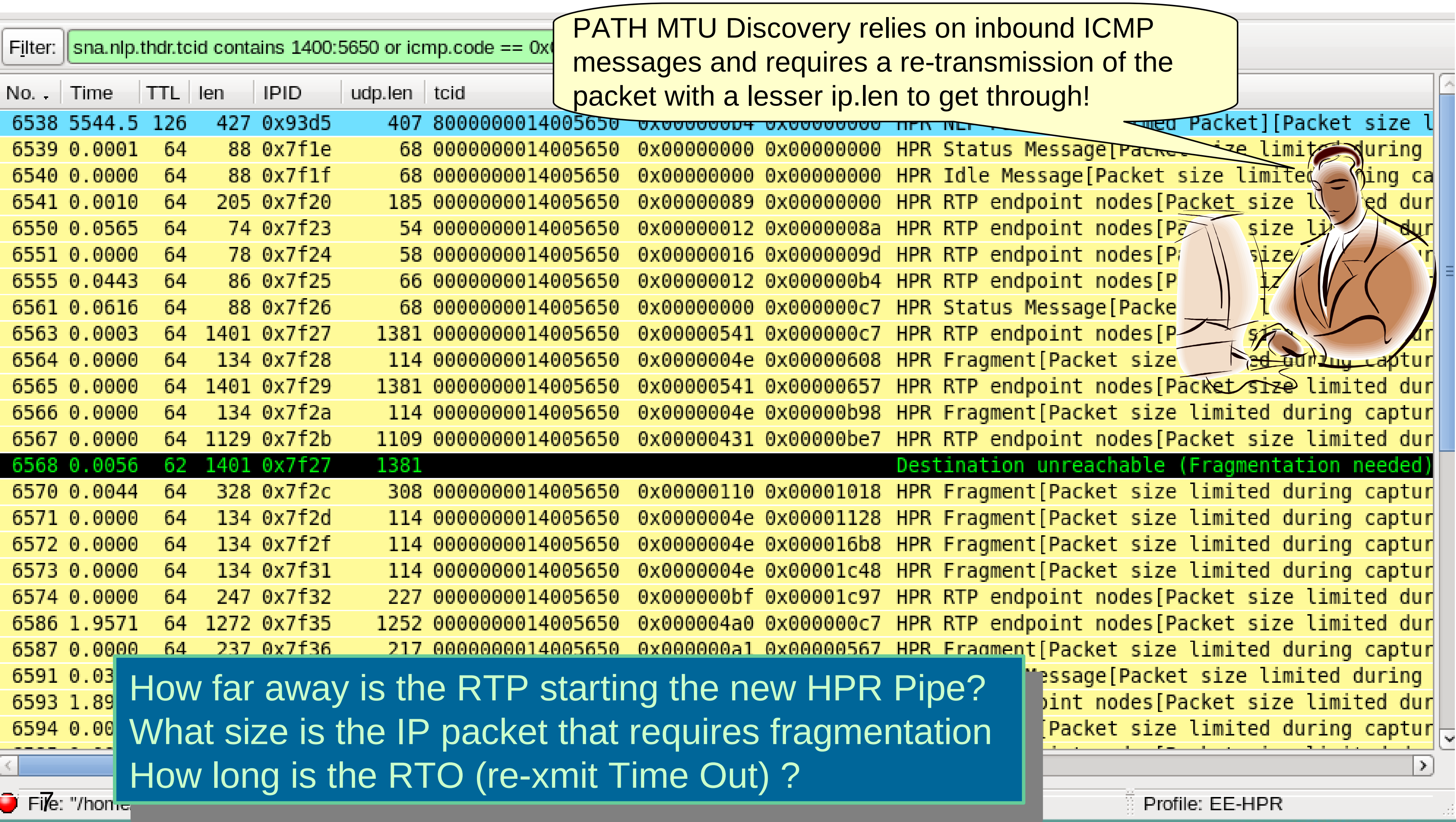
Knowing the initial TTL of a sending TCP stack allows you to guess its distance from the trace tool. Even though the initial TTL is a configurable option, most Operating Systems are using the default. Some common TTLs to remember are session are:



OS	ICMP	UDP	TCP
zOS	64	64	64
Linux		64	64
Tandem			64
Solaris	255	255	60
AIX	255	30	60
Win	128	128	128
i5		64	64
Routers	255	255	255

Warmup: EE PMTU Discovery in action

HPR Traffic filtered on an RTP pipe's TCID and ICMP msg



Filter: sna.nlp.thdr.tcid contains 1400:5650 or icmp.code == 0x

```
No. .  Time    TTL  len  IPID       udp.len  tcid
6538 5544.5 126   427 0x93d5      407 8000000014005650  0x000000b4 0x00000000 HPR NLP        med Packet][Packet size l
6539 0.0001  64    88 0x7f1e       68 0000000014005650  0x00000000 0x00000000 HPR Status Message[Packe   ize limit  during
6540 0.0000  64    88 0x7f1f       68 0000000014005650  0x00000000 0x00000000 HPR Idle Message[Packet size limite   ing ca
6541 0.0010  64   205 0x7f20      185 0000000014005650  0x00000089 0x00000000 HPR RTP endpoint nodes[Packet size l   ed dur
6550 0.0565  64    74 0x7f23       54 0000000014005650  0x00000012 0x0000008a HPR RTP endpoint nodes[Pa   size li    dur
6551 0.0000  64    78 0x7f24       58 0000000014005650  0x00000016 0x0000009d HPR RTP endpoint nodes[P   size
6555 0.0443  64    86 0x7f25       66 0000000014005650  0x00000012 0x000000b4 HPR RTP endpoint nodes[P
6561 0.0616  64    88 0x7f26       68 0000000014005650  0x00000000 0x000000c7 HPR Status Message[Packe
6563 0.0003  64  1401 0x7f27     1381 0000000014005650  0x00000541 0x000000c7 HPR RTP endpoint nodes[P
6564 0.0000  64   134 0x7f28      114 0000000014005650  0x0000004e 0x00000608 HPR Fragment[Packet size      ed during captur
6565 0.0000  64  1401 0x7f29     1381 0000000014005650  0x00000541 0x00000657 HPR RTP endpoint nodes[Packet size limited dur
6566 0.0000  64   134 0x7f2a      114 0000000014005650  0x0000004e 0x00000b98 HPR Fragment[Packet size limited during captur
6567 0.0000  64  1129 0x7f2b     1109 0000000014005650  0x00000431 0x00000be7 HPR RTP endpoint nodes[Packet size limited dur
6568 0.0056  62  1401 0x7f27     1381                                         Destination unreachable (Fragmentation needed)
6570 0.0044  64   328 0x7f2c      308 0000000014005650  0x00000110 0x00001018 HPR Fragment[Packet size limited during captur
6571 0.0000  64   134 0x7f2d      114 0000000014005650  0x0000004e 0x00001128 HPR Fragment[Packet size limited during captur
6572 0.0000  64   134 0x7f2f      114 0000000014005650  0x0000004e 0x000016b8 HPR Fragment[Packet size limited during captur
6573 0.0000  64   134 0x7f31      114 0000000014005650  0x0000004e 0x00001c48 HPR Fragment[Packet size limited during captur
6574 0.0000  64   247 0x7f32      227 0000000014005650  0x000000bf 0x00001c97 HPR RTP endpoint nodes[Packet size limited dur
6586 1.9571  64  1272 0x7f35     1252 0000000014005650  0x000004a0 0x000000c7 HPR RTP endpoint nodes[Packet size limited dur
6587 0.0000  64   237 0x7f36      217 0000000014005650  0x000000a1 0x00000567 HPR Fragment[Packet size limited during captur
6591 0.03                                                                    essage[Packet size limited during
6593 1.89                                                                    oint nodes[Packet size limited dur
6594 0.00                                                                    [Packet size limited during captur
```

File: "/home

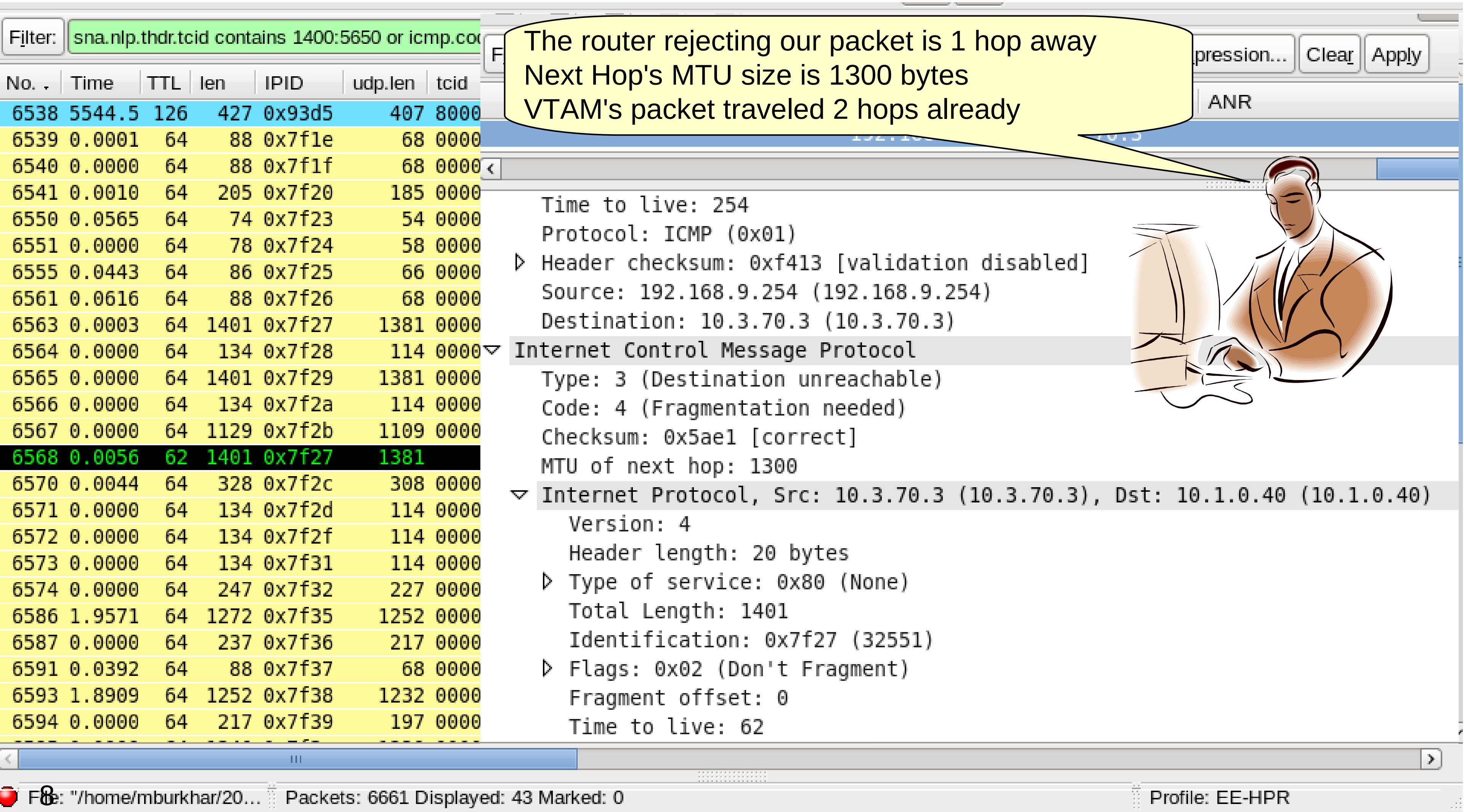
Profile: EE-HPR

PATH MTU Discovery relies on inbound ICMP messages and requires a re-transmission of the packet with a lesser ip.len to get through!

How far away is the RTP starting the new HPR Pipe?
What size is the IP packet that requires fragmentation
How long is the RTO (re-xmit Time Out) ?

Warmup: EE PMTUD in action - ICMP

ICMP message: Carries the original IP datagram ...



The router rejecting our packet is 1 hop away
Next Hop's MTU size is 1300 bytes
VTAM's packet traveled 2 hops already

```
No. .  Time    TTL  len  IPID    udp.len tcid
6538 5544.5  126   427 0x93d5     407 8000
6539 0.0001   64    88 0x7f1e      68 0000
6540 0.0000   64    88 0x7f1f      68 0000
6541 0.0010   64   205 0x7f20     185 0000
6550 0.0565   64    74 0x7f23      54 0000
6551 0.0000   64    78 0x7f24      58 0000
6555 0.0443   64    86 0x7f25      66 0000
6561 0.0616   64    88 0x7f26      68 0000
6563 0.0003   64  1401 0x7f27    1381 0000
6564 0.0000   64   134 0x7f28     114 0000
6565 0.0000   64  1401 0x7f29    1381 0000
6566 0.0000   64   134 0x7f2a     114 0000
6567 0.0000   64  1129 0x7f2b    1109 0000
6568 0.0056   62  1401 0x7f27    1381
6570 0.0044   64   328 0x7f2c     308 0000
6571 0.0000   64   134 0x7f2d     114 0000
6572 0.0000   64   134 0x7f2f     114 0000
6573 0.0000   64   134 0x7f31     114 0000
6574 0.0000   64   247 0x7f32     227 0000
6586 1.9571   64  1272 0x7f35    1252 0000
6587 0.0000   64   237 0x7f36     217 0000
6591 0.0392   64    88 0x7f37      68 0000
6593 1.8909   64  1252 0x7f38    1232 0000
6594 0.0000   64   217 0x7f39     197 0000
```

```
    Time to live: 254
    Protocol: ICMP (0x01)
  ▷ Header checksum: 0xf413 [validation disabled]
    Source: 192.168.9.254 (192.168.9.254)
    Destination: 10.3.70.3 (10.3.70.3)
▽ Internet Control Message Protocol
    Type: 3 (Destination unreachable)
    Code: 4 (Fragmentation needed)
    Checksum: 0x5ae1 [correct]
    MTU of next hop: 1300
  ▽ Internet Protocol, Src: 10.3.70.3 (10.3.70.3), Dst: 10.1.0.40 (10.1.0.40)
      Version: 4
      Header length: 20 bytes
    ▷ Type of service: 0x80 (None)
      Total Length: 1401
      Identification: 0x7f27 (32551)
    ▷ Flags: 0x02 (Don't Fragment)
      Fragment offset: 0
      Time to live: 62
```

File: "/home/mburkhar/20... Packets: 6661 Displayed: 43 Marked: 0 Profile: EE-HPR

Background information – 3-way Handshake

http://en.wikipedia.org/wiki/3_way_handshake

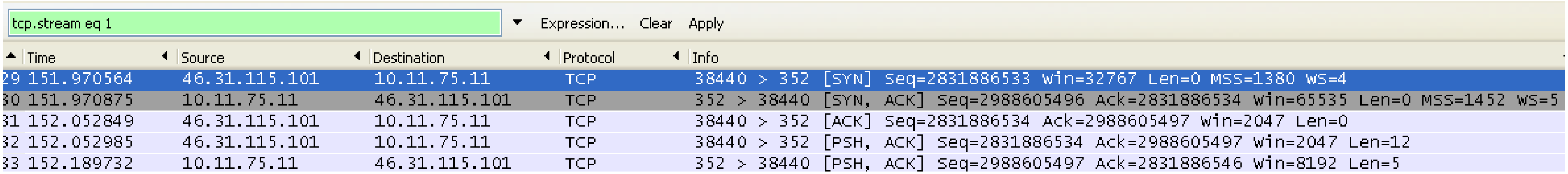
The “three-way handshake” describes the flow of a new TCP connection. It takes 3 packets to successfully establish a session between two TCP endpoints.

- Client sending a SYN packet (SYN Flag is set in TCP header)
- Server sending a SYN_ACK (SYN and ACK Flag is set in TCP header)
- Client sending an ACK to acknowledge the receipt of the server's SYN_ACK

In the SYN packet(s) the TCP stacks exchange parameters on how the characteristics of the session should be. These are exchanged using TCP options like MSS and Window Scaling, SACK and Timestamp options. The options need to be padded to a 4 word boundary using the NOP option.
The presence of some TCP options, their values and the sequence of their appearance form a TCP fingerprint that can be used to detect the sender's operating system.
See: http://en.wikipedia.org/wiki/TCP/IP_stack_fingerprinting
In addition, it is important to look at the delta time between the packets to identify where the trace was taken and was the Round Trip Time (RTT) on the connection is.

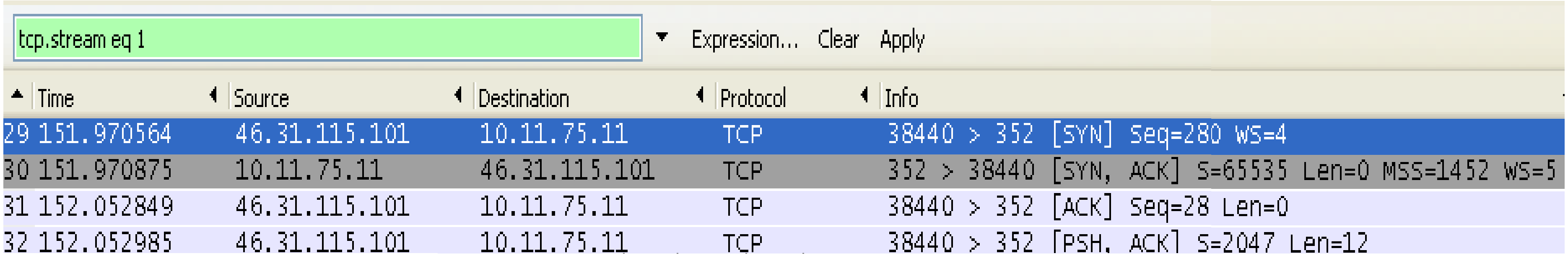
3-way handshake – Round Trip Time

Who is who? Who is where? Where is the trace taken?



tcp.stream eq 1

	Time	Source	Destination	Protocol	Info
29	151.970564	46.31.115.101	10.11.75.11	TCP	38440 > 352 [SYN] Seq=2831886533 Win=32767 Len=0 MSS=1380 WS=4
30	151.970875	10.11.75.11	46.31.115.101	TCP	352 > 38440 [SYN, ACK] Seq=2988605496 Ack=2831886534 Win=65535 Len=0 MSS=1452 WS=5
31	152.052849	46.31.115.101	10.11.75.11	TCP	38440 > 352 [ACK] Seq=2831886534 Ack=2988605497 Win=2047 Len=0
32	152.052985	46.31.115.101	10.11.75.11	TCP	38440 > 352 [PSH, ACK] Seq=2831886534 Ack=2988605497 Win=2047 Len=12
33	152.189732	10.11.75.11	46.31.115.101	TCP	352 > 38440 [PSH, ACK] Seq=2988605497 Ack=2831886546 Win=8192 Len=5



tcp.stream eq 1

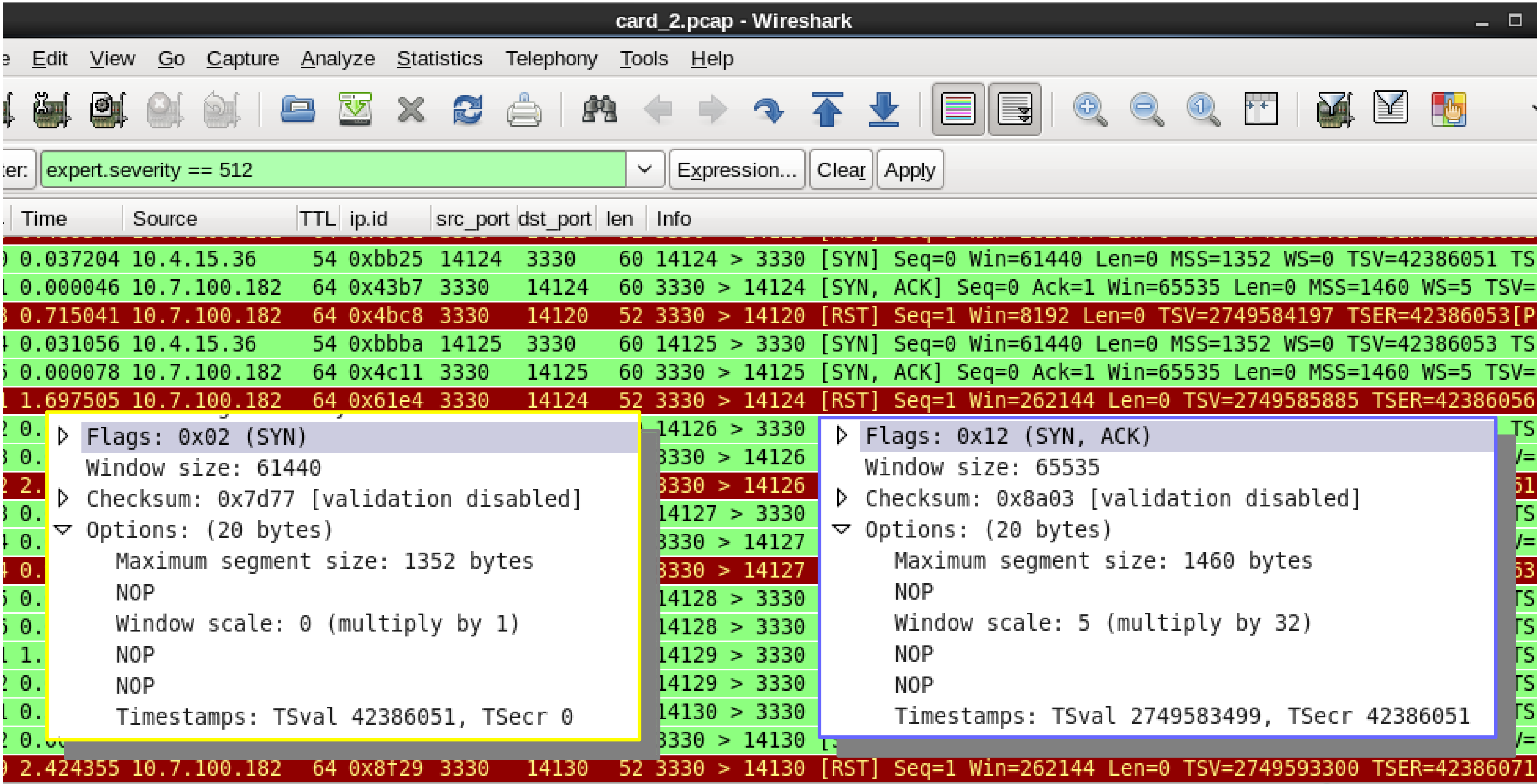
	Time	Source	Destination	Protocol	Info
29	151.970564	46.31.115.101	10.11.75.11	TCP	38440 > 352 [SYN] Seq=280 WS=4
30	151.970875	10.11.75.11	46.31.115.101	TCP	352 > 38440 [SYN, ACK] S=65535 Len=0 MSS=1452 WS=5
31	152.052849	46.31.115.101	10.11.75.11	TCP	38440 > 352 [ACK] Seq=28 Len=0
32	152.052985	46.31.115.101	10.11.75.11	TCP	38440 > 352 [PSH, ACK] S=2047 Len=12

It's always worthwhile to look at the delta time between packets. It helps to identify, where the trace was taken

Who is the client, who is the server?
Where was the trace taken?
What is the “Round Trip Time” ?

3-way handshake – p0f fingerprints

Who is who? Who is where? Where is the trace taken?



card_2.pcap - Wireshark
Edit View Go Capture Analyze Statistics Telephony Tools Help
Expression... Clear Apply

Filter: expert.severity == 512

```
Time     Source       TTL ip.id  src_port dst_port len Info
0.037204 10.4.15.36    54 0xbb25 14124    3330     60 14124 > 3330 [SYN] Seq=0 Win=61440 Len=0 MSS=1352 WS=0 TSV=42386051 TS
0.000046 10.7.100.182  64 0x43b7 3330     14124    60 3330 > 14124 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=5 TSV=
0.715041 10.7.100.182  64 0x4bc8 3330     14120    52 3330 > 14120 [RST] Seq=1 Win=8192 Len=0 TSV=2749584197 TSER=42386053[P
0.031056 10.4.15.36    54 0xbbba 14125    3330     60 14125 > 3330 [SYN] Seq=0 Win=61440 Len=0 MSS=1352 WS=0 TSV=42386053 TS
0.000078 10.7.100.182  64 0x4c11 3330     14125    60 3330 > 14125 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=5 TSV=
1.697505 10.7.100.182  64 0x61e4 3330     14124    52 3330 > 14124 [RST] Seq=1 Win=262144 Len=0 TSV=2749585885 TSER=42386056
0.                                                   14126 > 3330                                                         TS
0.                                                   3330 > 14126                                                         V=
2.                                                   3330 > 14126                                                         51
0.                                                   14127 > 3330                                                         TS
0.                                                   3330 > 14127                                                         V=
0.                                                   3330 > 14127                                                         53
0.                                                   14128 > 3330                                                         TS
0.                                                   14128 > 3330                                                         TS
1.                                                   14129 > 3330                                                         TS
0.                                                   14129 > 3330                                                         TS
0.                                                   14130 > 3330                                                         TS
0.                                                   3330 > 14130                                                         V=
2.424355 10.7.100.182  64 0x8f29 3330     14130    52 3330 > 14130 [RST] Seq=1 Win=262144 Len=0 TSV=2749593300 TSER=42386071
```

```
▷ Flags: 0x02 (SYN)
  Window size: 61440
▷ Checksum: 0x7d77 [validation disabled]
▽ Options: (20 bytes)
    Maximum segment size: 1352 bytes
    NOP
    Window scale: 0 (multiply by 1)
    NOP
    NOP
    Timestamps: TSval 42386051, TSecr 0
```

```
▷ Flags: 0x12 (SYN, ACK)
  Window size: 65535
▷ Checksum: 0x8a03 [validation disabled]
▽ Options: (20 bytes)
    Maximum segment size: 1460 bytes
    NOP
    Window scale: 5 (multiply by 32)
    NOP
    NOP
    Timestamps: TSval 2749583499, TSecr 42386051
```

How to convert SYSTCPDA to tcpdump

OPTIONS((SNIFFER(size TCPDUMP)))

```
//BURKSNIF JOB (7904,NCS),BURKHAR,MSGLEVEL=(1,1),MSGCLASS=K,CLASS=A,
//    NOTIFY=&SYSUID.,REGION=0M,TIME=150
//    SET INDUMP='ONTOP.TCPIP.DBUG2012.MQRESET.CSDUMP'
//     SET SNIFF='ONTOP.TCPIP.DBUG2012.MQRESET.PCAP'
//    SET MIGLIB='TOP.ZOSR1D.MIGLIB'
//*             OTHERWISE THE SNIFFER FILE WILL BE EMPTY|||
//*  THIS JOB CONVERTS A PACKET TRACE TO SNIFFER
//*  ATTENTION: PLEASE VERIFY THE TCPIP JOBNAME IS CORRECT
//*             OTHERWISE THE SNIFFER FILE WILL BE EMPTY|||
//IPCSBTCH EXEC PGM=IKJEFT01,DYNAMNBR=30
//STEPLIB  DD DISP=SHR,DSN=&MIGLIB.
//IPCSDDIR DD DISP=SHR,DSN=&SYSUID..ZOS1B.DIRECTRY
//IPCSDUMP DD *
//SYSTSPRT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//INDMP    DD DISP=SHR,DSN=&INDUMP.
//SNIFFER  DD DSN=&SNIFF.,
//  DISP=(NEW,CATLG),LRECL=1560,SPACE=(CYL,(550,50)),RECFM=VB,DSORG=PS
//* DISP=SHR
//IPCSPRNT DD SYSOUT=*
//IPCSTOC  DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSTSIN  DD *
 PROFILE MSGID
  IPCS NOPARM
  SETD PRINT NOTERM LENGTH(160000) NOCONFIRM FILE(INDMP)
  DROPD
 CTRACE COMP(SYSTCPDA) SUB((TCPIP)) -
    OPTIONS((SNIFFER(1514 TCPDUMP)))
 END
```

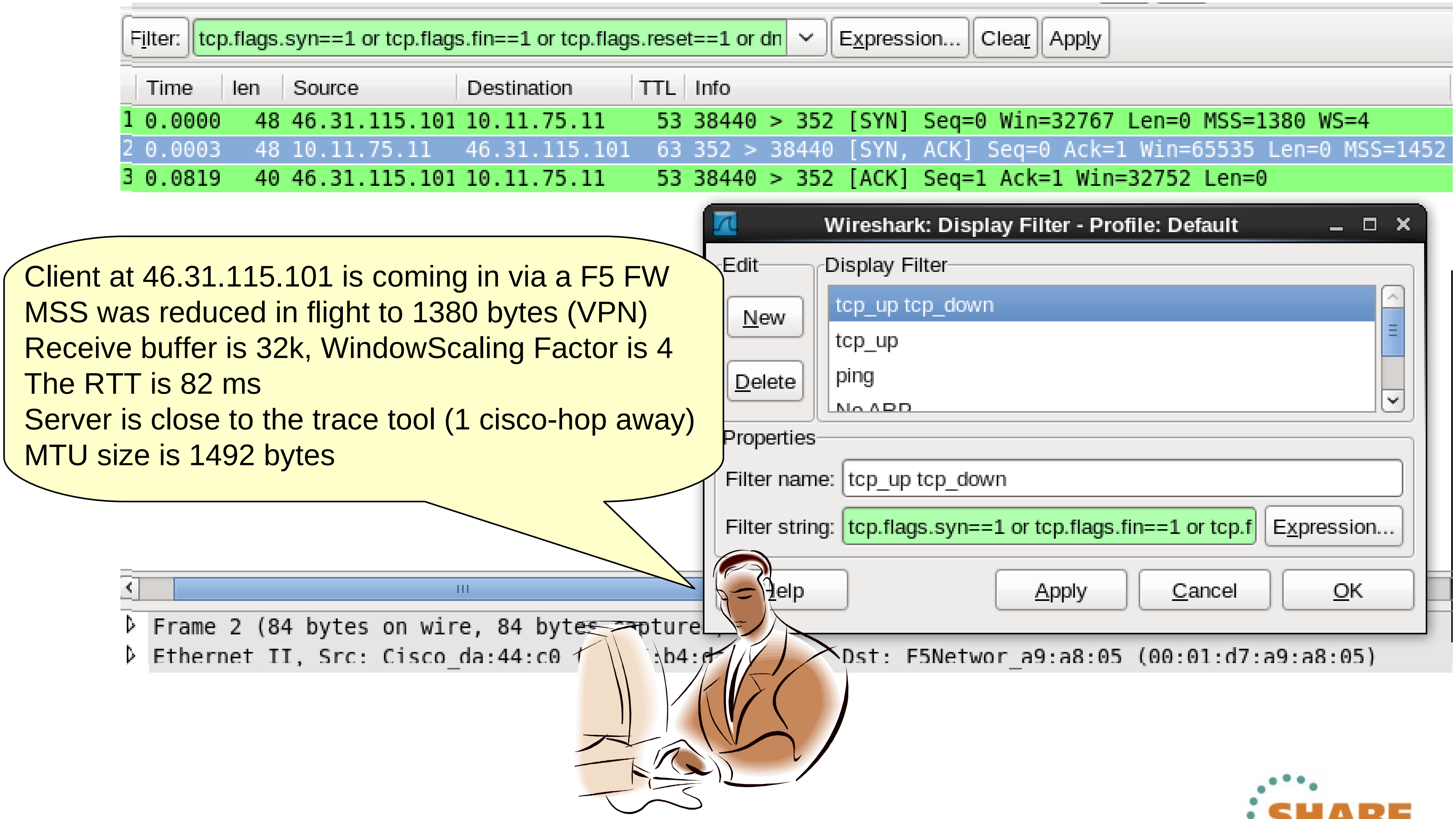
Takes an external CTRACE writer file or a Dump
Can process SYSTCPDA and SYSTCPOT
Can abbreviate during export

Solving Network Problems – 3 Step Process

- Understand the Topology
  - What Platforms are involved?
  - What does the Network Infrastructure look like?
  - What parameters are configured and where?
- Understand Problem
  - What is the concern?
  - What is the impact?
  - What is the root cause?
- Evaluate possible Solutions
  - Ease of implementation
  - Scope of responsibility

Problem 1: Performance Problem – Part I.

What is configured? How does the network look like?



Filter: tcp.flags.syn==1 or tcp.flags.fin==1 or tcp.flags.reset==1 or dn

```
  Time    len Source        Destination   TTL Info
1 0.0000  48 46.31.115.101 10.11.75.11    53 38440 > 352 [SYN] Seq=0 Win=32767 Len=0 MSS=1380 WS=4
2 0.0003  48 10.11.75.11   46.31.115.101  63 352 > 38440 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1452
3 0.0819  40 46.31.115.101 10.11.75.11    53 38440 > 352 [ACK] Seq=1 Ack=1 Win=32752 Len=0
```

Filter name: tcp_up tcp_down

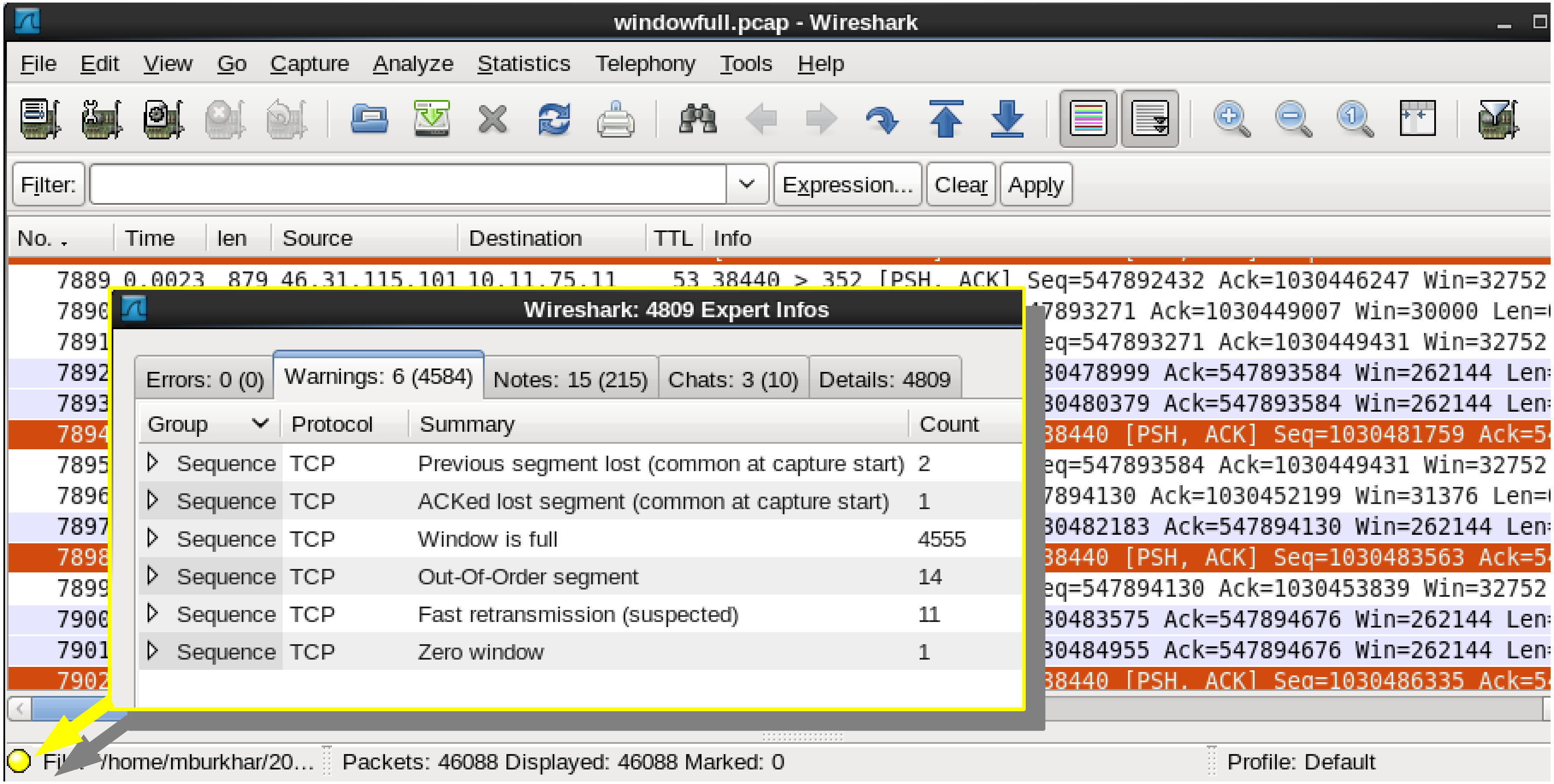
Filter string: tcp.flags.syn==1 or tcp.flags.fin==1 or tcp.f

```
▷ Frame 2 (84 bytes on wire, 84 bytes capture
▷ Ethernet II, Src: Cisco_da:44:c0 (      :b4:d      Dst: F5Networ_a9:a8:05 (00:01:d7:a9:a8:05)
```

Client at 46.31.115.101 is coming in via a F5 FW
MSS was reduced in flight to 1380 bytes (VPN)
Receive buffer is 32k, WindowScaling Factor is 4
The RTT is 82 ms
Server is close to the trace tool (1 cisco-hop away)
MTU size is 1492 bytes

Problem 1: Performance Problem – Part II.

Expert Infos: 4555 “Window is full” messages in the trace


windowfull.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply

```
No.    Time    len  Source          Destination   TTL  Info
  7889 0.0023  879 46.31.115.101 10.11.75.11      53 38440 > 352 [PSH, ACK] Seq=547892432 Ack=1030446247 Win=32752
  7890                                                  7893271 Ack=1030449007 Win=30000 Len=
  7891                                                  eq=547893271 Ack=1030449431 Win=32752
  7892                                                  30478999 Ack=547893584 Win=262144 Len=
  7893                                                  30480379 Ack=547893584 Win=262144 Len=
  7894                                                  38440 [PSH, ACK] Seq=1030481759 Ack=5
  7895                                                  eq=547893584 Ack=1030449431 Win=32752
  7896                                                  7894130 Ack=1030452199 Win=31376 Len=
  7897                                                  30482183 Ack=547894130 Win=262144 Len=
  7898                                                  38440 [PSH, ACK] Seq=1030483563 Ack=5
  7899                                                  eq=547894130 Ack=1030453839 Win=32752
  7900                                                  30483575 Ack=547894676 Win=262144 Len=
  7901                                                  30484955 Ack=547894676 Win=262144 Len=
  7902                                                  38440 [PSH, ACK] Seq=1030486335 Ack=5
```

Wireshark: 4809 Expert Infos

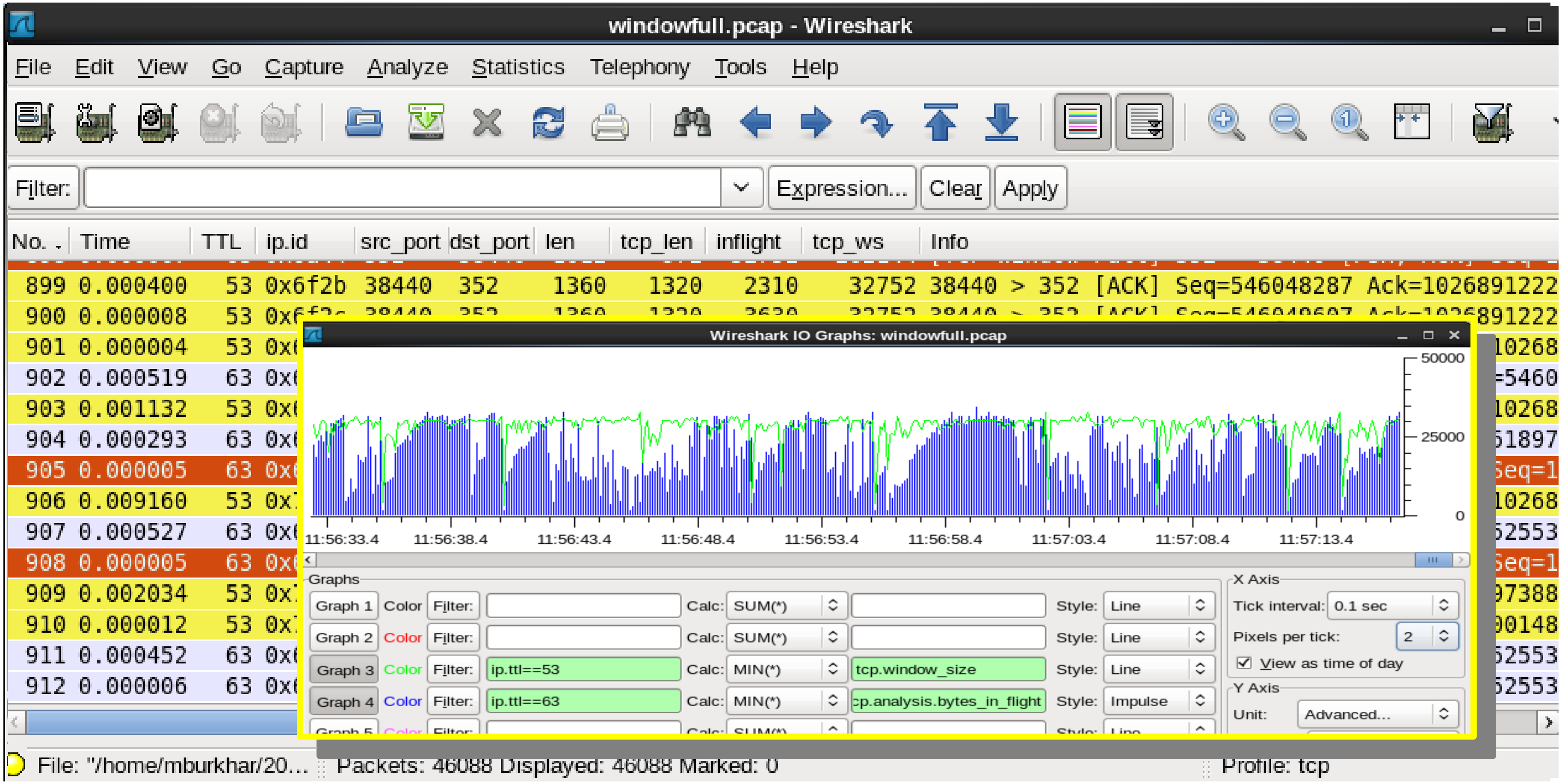
Errors: 0 (0) | Warnings: 6 (4584) | Notes: 15 (215) | Chats: 3 (10) | Details: 4809

Group	Protocol	Summary	Count
▷ Sequence	TCP	Previous segment lost (common at capture start)	2
▷ Sequence	TCP	ACKed lost segment (common at capture start)	1
▷ Sequence	TCP	Window is full	4555
▷ Sequence	TCP	Out-Of-Order segment	14
▷ Sequence	TCP	Fast retransmission (suspected)	11
▷ Sequence	TCP	Zero window	1

Fi /home/mburkhar/20… Packets: 46088 Displayed: 46088 Marked: 0 Profile: Default

Problem 1: Performance Problem – Part III.

IO Graph: bytes_inflight and tcp_ws over time



windowfull.pcap - Wireshark

File Edit View Go Capture Analyze Statistics Telephony Tools Help
Filter:
Expression... Clear Apply

```
No. .  Time      TTL ip.id  src_port dst_port len  tcp_len inflight tcp_ws Info
899 0.000400  53 0x6f2b 38440    352      1360 1320    2310     32752 38440 > 352 [ACK] Seq=546048287 Ack=1026891222
900 0.000008  53 0x6f2c 38440    352      1360 1320    3630     32752 38440 > 352 [ACK] Seq=546049607 Ack=1026891222
901 0.000004  53 0x                                                                                          10268
902 0.000519  63 0x                                                                                          =5460
903 0.001132  53 0x                                                                                          10268
904 0.000293  63 0x                                                                                          51897
905 0.000005  63 0x                                                                                          Seq=1
906 0.009160  53 0x                                                                                          10268
907 0.000527  63 0x                                                                                          52553
908 0.000005  63 0x                                                                                          Seq=1
909 0.002034  53 0x                                                                                          97388
910 0.000012  53 0x                                                                                          00148
911 0.000452  63 0x                                                                                          52553
912 0.000006  63 0x                                                                                          52553
```

Wireshark IO Graphs: windowfull.pcap

Graphs

Graph 1	Color	Filter:		Calc: SUM(*)		Style: Line
Graph 2	Color	Filter:		Calc: SUM(*)		Style: Line
Graph 3	Color	Filter:	ip.ttl==53	Calc: MIN(*)	tcp.window_size	Style: Line
Graph 4	Color	Filter:	ip.ttl==63	Calc: MIN(*)	cp.analysis.bytes_in_flight	Style: Impulse
Graph 5	Color	Filter:		Calc: SUM(*)		Style: Line

X Axis

Tick interval: 0.1 sec

Pixels per tick: 2

☑ View as time of day

Y Axis

Unit: Advanced...

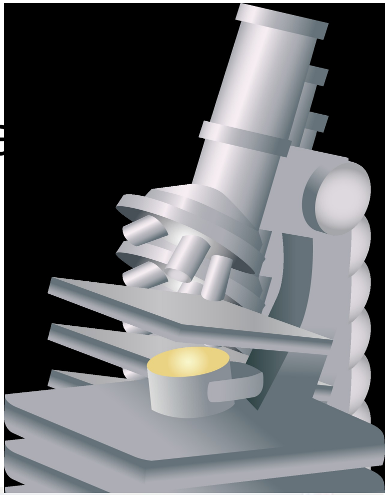
File: "/home/mburkhar/20... Packets: 46088 Displayed: 46088 Marked: 0 Profile: tcp

Problem 1: Performance Problem - Solution

- Understand the Topology
  - What Platforms are involved? z/OS and ???
  - What does the Network Infrastructure look like? 80ms RTT
  - What parameters are configured and where? 32K receivebuf
- Understand Problem
  - What is the concern? Poor Performance on TCP session
  - What is the impact? Number of transactions/s reduced
  - What is the root cause? TCP session stalls waiting for window
- Evaluate possible Solutions
  - Ease of implementation Increase TCP receive buffer at client
  - Scope of responsibility Business Partner

Problem 2: intermittent ABENDs in CICS

- Understand Problem
  - What is the concern?
    - CICS ABENDs when expected data is not completely read from socket
  - What is the impact?
    - CICS transactions abort, the socket is close
    - Clients need to reconnect
  - What is the root cause?

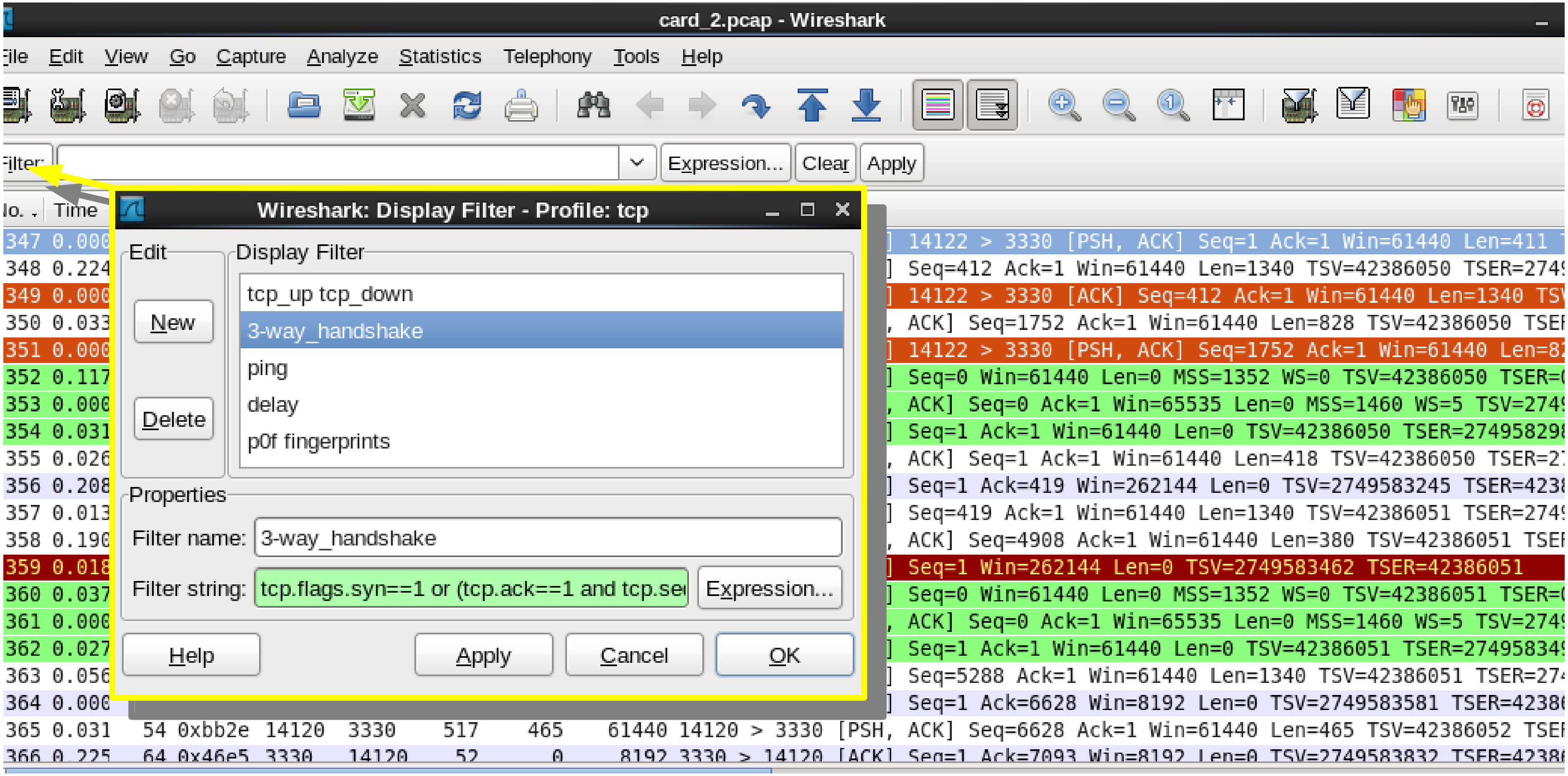


WIRESHARK

Network Protocol Analyzer

Problem 2: CICS ABEND – Part I.

The Topology: 3-way_handshake filter



Display Filter:
- tcp_up tcp_down
- 3-way_handshake
- ping
- delay
- p0f fingerprints

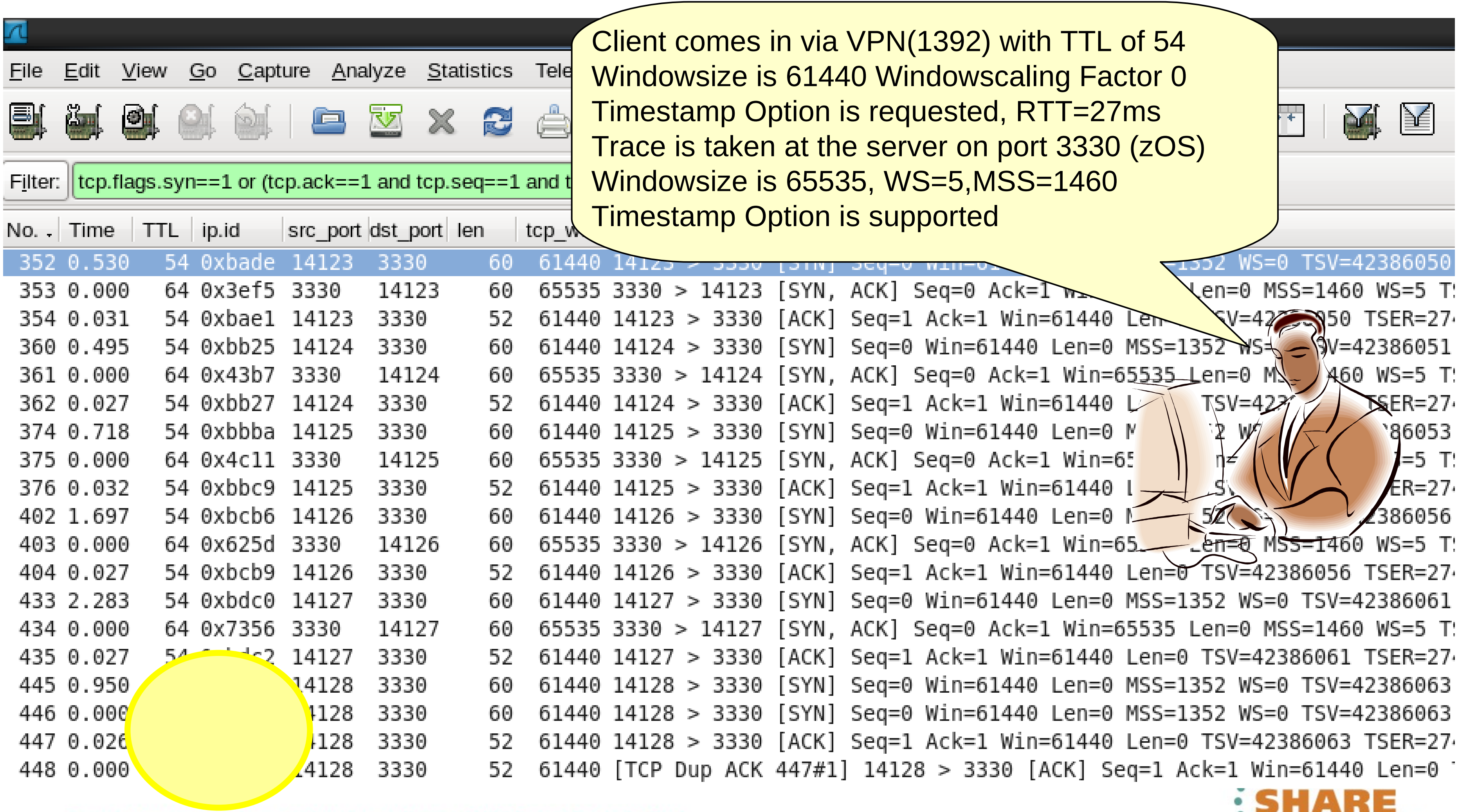
Filter name: 3-way_handshake

Filter string: tcp.flags.syn==1 or (tcp.ack==1 and tcp.se

```
347 0.000   ] 14122 > 3330 [PSH, ACK] Seq=1 Ack=1 Win=61440 Len=411 T
348 0.224   ] Seq=412 Ack=1 Win=61440 Len=1340 TSV=42386050 TSER=274
349 0.000   ] 14122 > 3330 [ACK] Seq=412 Ack=1 Win=61440 Len=1340 TSV
350 0.033   , ACK] Seq=1752 Ack=1 Win=61440 Len=828 TSV=42386050 TSEF
351 0.000   ] 14122 > 3330 [PSH, ACK] Seq=1752 Ack=1 Win=61440 Len=82
352 0.117   ] Seq=0 Win=61440 Len=0 MSS=1352 WS=0 TSV=42386050 TSER=
353 0.000   , ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=5 TSV=274
354 0.031   ] Seq=1 Ack=1 Win=61440 Len=0 TSV=42386050 TSER=27495829
355 0.026   , ACK] Seq=1 Ack=1 Win=61440 Len=418 TSV=42386050 TSER=2
356 0.208   ] Seq=1 Ack=419 Win=262144 Len=0 TSV=2749583245 TSER=4238
357 0.013   ] Seq=419 Ack=1 Win=61440 Len=1340 TSV=42386051 TSER=274
358 0.190   , ACK] Seq=4908 Ack=1 Win=61440 Len=380 TSV=42386051 TSEF
359 0.018   ] Seq=1 Win=262144 Len=0 TSV=2749583462 TSER=42386051
360 0.037   ] Seq=0 Win=61440 Len=0 MSS=1352 WS=0 TSV=42386051 TSER=
361 0.000   , ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=5 TSV=274
362 0.027   ] Seq=1 Ack=1 Win=61440 Len=0 TSV=42386051 TSER=27495834
363 0.056   ] Seq=5288 Ack=1 Win=61440 Len=1340 TSV=42386051 TSER=27
364 0.000   ] Seq=1 Ack=6628 Win=8192 Len=0 TSV=2749583581 TSER=4238
365 0.031  54 0xbb2e  14120  3330   517   465   61440 14120 > 3330 [PSH, ACK] Seq=6628 Ack=1 Win=61440 Len=465 TSV=42386052 TSEF
366 0.225  64 0x46e5  3330   14120   52     0    8192 3330 > 14120 [ACK] Seq=1 Ack=7093 Win=8192 Len=0 TSV=2749583832 TSER=4238
```

Problem 2: CICS ABEND – Part I.

The Topology: 3-way-Handshakes



Filter: tcp.flags.syn==1 or (tcp.ack==1 and tcp.seq==1 and t

```
No. .  Time   TTL  ip.id    src_port dst_port len    tcp_w
352 0.530   54 0xbade 14123   3330      60  61440 14123 > 3330 [SYN] Seq=0 Win=61            1352 WS=0 TSV=42386050
353 0.000   64 0x3ef5 3330    14123     60  65535 3330 > 14123 [SYN, ACK] Seq=0 Ack=1 Wi         Len=0 MSS=1460 WS=5 T
354 0.031   54 0xbae1 14123   3330      52  61440 14123 > 3330 [ACK] Seq=1 Ack=1 Win=61440 Len      SV=42    50 TSER=27
360 0.495   54 0xbb25 14124   3330      60  61440 14124 > 3330 [SYN] Seq=0 Win=61440 Len=0 MSS=1352 WS=   SV=42386051
361 0.000   64 0x43b7 3330    14124     60  65535 3330 > 14124 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 M    460 WS=5 T
362 0.027   54 0xbb27 14124   3330      52  61440 14124 > 3330 [ACK] Seq=1 Ack=1 Win=61440 L       TSV=42          SER=27
374 0.718   54 0xbbba 14125   3330      60  61440 14125 > 3330 [SYN] Seq=0 Win=61440 Len=0 M         2 W         86053
375 0.000   64 0x4c11 3330    14125     60  65535 3330 > 14125 [SYN, ACK] Seq=0 Ack=1 Win=65       n=            =5 T
376 0.032   54 0xbbc9 14125   3330      52  61440 14125 > 3330 [ACK] Seq=1 Ack=1 Win=61440 L       S            ER=27
402 1.697   54 0xbcb6 14126   3330      60  61440 14126 > 3330 [SYN] Seq=0 Win=61440 Len=0 M       52            386056
403 0.000   64 0x625d 3330    14126     60  65535 3330 > 14126 [SYN, ACK] Seq=0 Ack=1 Win=65      Len=0 MSS=1460 WS=5 T
404 0.027   54 0xbcb9 14126   3330      52  61440 14126 > 3330 [ACK] Seq=1 Ack=1 Win=61440 Len=0 TSV=42386056 TSER=27
433 2.283   54 0xbdc0 14127   3330      60  61440 14127 > 3330 [SYN] Seq=0 Win=61440 Len=0 MSS=1352 WS=0 TSV=42386061
434 0.000   64 0x7356 3330    14127     60  65535 3330 > 14127 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=5 T
435 0.027   5        c2 14127   3330      52  61440 14127 > 3330 [ACK] Seq=1 Ack=1 Win=61440 Len=0 TSV=42386061 TSER=27
445 0.950                4128   3330      60  61440 14128 > 3330 [SYN] Seq=0 Win=61440 Len=0 MSS=1352 WS=0 TSV=42386063
446 0.00                 128   3330      60  61440 14128 > 3330 [SYN] Seq=0 Win=61440 Len=0 MSS=1352 WS=0 TSV=42386063
447 0.02                 128   3330      52  61440 14128 > 3330 [ACK] Seq=1 Ack=1 Win=61440 Len=0 TSV=42386063 TSER=27
448 0.000                4128   3330      52  61440 [TCP Dup ACK 447#1] 14128 > 3330 [ACK] Seq=1 Ack=1 Win=61440 Len=0
```

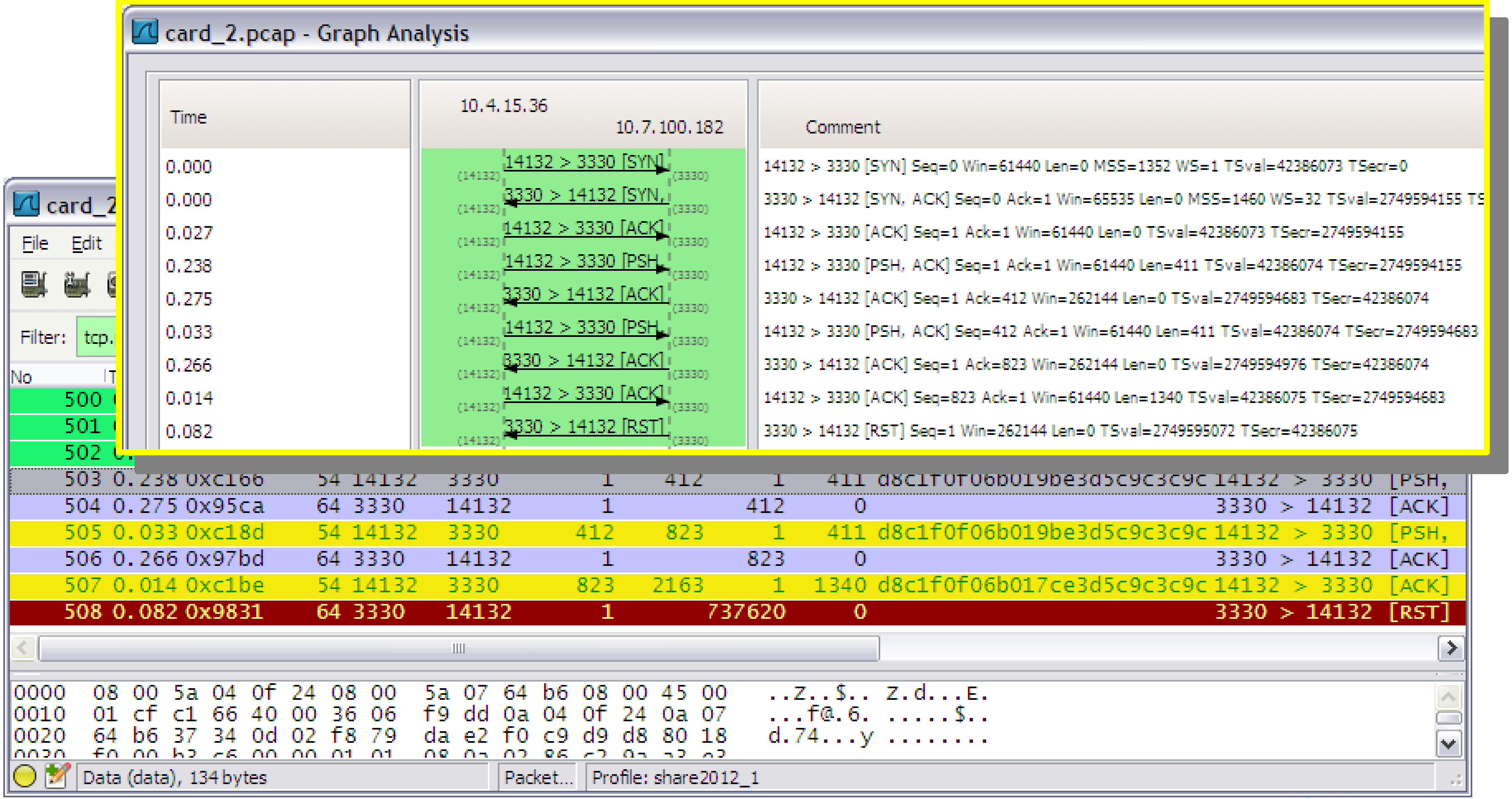
Client comes in via VPN(1392) with TTL of 54
Windowsize is 61440 Windowscaling Factor 0
Timestamp Option is requested, RTT=27ms
Trace is taken at the server on port 3330 (zOS)
Windowsize is 65535, WS=5,MSS=1460
Timestamp Option is supported

Problem 2: CICS ABEND – Topology

- Understand the Topology
  - The trace was taken on a Sysplex Distributor stack
  - The server is running on z/OS
    - MSS = 1452 → MTU size is 1492
    - Window Scaling factor is 5, multiply by 32
    - The advertized windowsize is 65535
  - The client is 10 hops away from the trace tool
    - Window Scaling factor is 0, multiply by 1
    - offered TCP Window Size is 61440 (Tandem)
    - The RTT is 27 ms
  - The available MTU size end to end is 1392 bytes
    - A VPN Tunnel is being used requiring additional headers for IPSec encryption/authentication

Problem 2: CICS ABEND – Flow Chart

Filter on client port, Statistics → Flowchart



```
Time     10.4.15.36          10.7.100.182     Comment
0.000    (14132) 14132 > 3330 [SYN]    (3330)   14132 > 3330 [SYN] Seq=0 Win=61440 Len=0 MSS=1352 WS=1 TSval=42386073 TSecr=0
0.000    (14132) 3330 > 14132 [SYN,    (3330)   3330 > 14132 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460 WS=32 TSval=2749594155 TS
0.027    (14132) 14132 > 3330 [ACK]    (3330)   14132 > 3330 [ACK] Seq=1 Ack=1 Win=61440 Len=0 TSval=42386073 TSecr=2749594155
0.238    (14132) 14132 > 3330 [PSH,    (3330)   14132 > 3330 [PSH, ACK] Seq=1 Ack=1 Win=61440 Len=411 TSval=42386074 TSecr=2749594155
0.275    (14132) 3330 > 14132 [ACK]    (3330)   3330 > 14132 [ACK] Seq=1 Ack=412 Win=262144 Len=0 TSval=2749594683 TSecr=42386074
0.033    (14132) 14132 > 3330 [PSH,    (3330)   14132 > 3330 [PSH, ACK] Seq=412 Ack=1 Win=61440 Len=411 TSval=42386074 TSecr=2749594683
0.266    (14132) 3330 > 14132 [ACK]    (3330)   3330 > 14132 [ACK] Seq=1 Ack=823 Win=262144 Len=0 TSval=2749594976 TSecr=42386074
0.014    (14132) 14132 > 3330 [ACK]    (3330)   14132 > 3330 [ACK] Seq=823 Ack=1 Win=61440 Len=1340 TSval=42386075 TSecr=2749594683
0.082    (14132) 3330 > 14132 [RST]    (3330)   3330 > 14132 [RST] Seq=1 Win=262144 Len=0 TSval=2749595072 TSecr=42386075
```

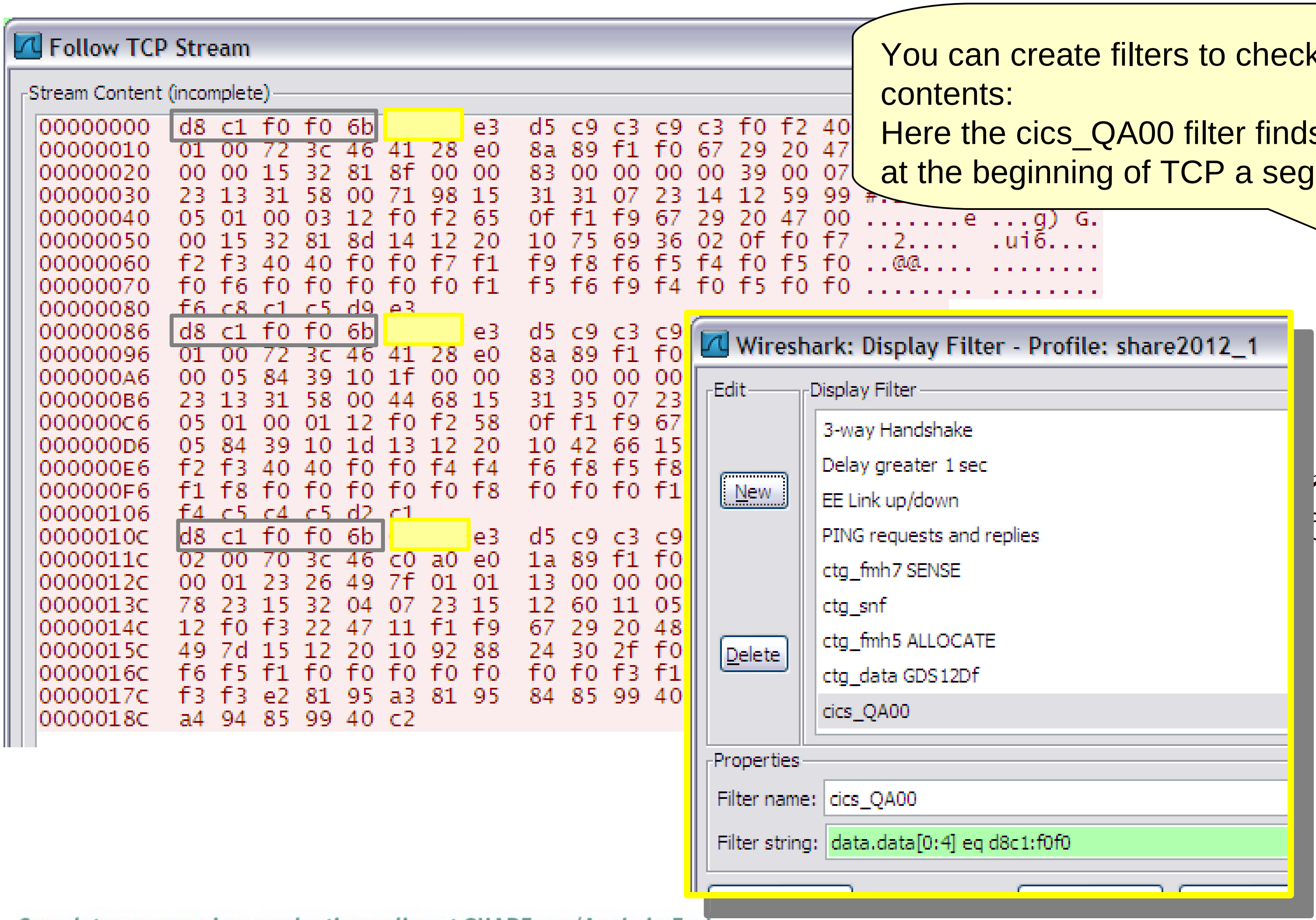
```
 500 (
 501 (
 502 0.
 503 0.238 0xc166   54 14132  3330      1     412        1   411 d8c1f0f06b019be3d5c9c3c9c 14132 > 3330 [PSH,
 504 0.275 0x95ca   64 3330   14132     1               412     0                           3330 > 14132 [ACK]
 505 0.033 0xc18d   54 14132  3330    412     823        1   411 d8c1f0f06b019be3d5c9c3c9c 14132 > 3330 [PSH,
 506 0.266 0x97bd   64 3330   14132     1               823     0                           3330 > 14132 [ACK]
 507 0.014 0xc1be   54 14132  3330    823    2163        1  1340 d8c1f0f06b017ce3d5c9c3c9c 14132 > 3330 [ACK]
 508 0.082 0x9831   64 3330   14132     1            737620     0                           3330 > 14132 [RST]
```

```
0000  08 00 5a 04 0f 24 08 00  5a 07 64 b6 08 00 45 00   ..Z..$.. Z.d...E.
0010  01 cf c1 66 40 00 36 06  f9 dd 0a 04 0f 24 0a 07   ...f@.6. .....$..
0020  64 b6 37 34 0d 02 f8 79  da e2 f0 c9 d9 d8 80 18   d.74...y ........
```

22 Complete your sessions evaluation online at SHARE.org/AnaheimEval

Problem 2: CICS ABEND – filter on data

Follow TCP Stream: Data Structure



Follow TCP Stream

Stream Content (incomplete)

```
00000000  d8 c1 f0 f0 6b       e3  d5 c9 c3 c9 c3 f0 f2 40
00000010  01 00 72 3c 46 41 28 e0  8a 89 f1 f0 67 29 20 47
00000020  00 00 15 32 81 8f 00 00  83 00 00 00 00 39 00 07
00000030  23 13 31 58 00 71 98 15  31 31 07 23 14 12 59 99 #.
00000040  05 01 00 03 12 f0 f2 65  0f f1 f9 67 29 20 47 00 .......e ...g) G.
00000050  00 15 32 81 8d 14 12 20  10 75 69 36 02 0f f0 f7 ..2....  .ui6....
00000060  f2 f3 40 40 f0 f0 f7 f1  f9 f8 f6 f5 f4 f0 f5 f0 ..@@.... ........
00000070  f0 f6 f0 f0 f0 f0 f0 f1  f5 f6 f9 f4 f0 f5 f0 f0 ........ ........
00000080  f6 c8 c1 c5 d9 e3
00000086  d8 c1 f0 f0 6b       e3  d5 c9 c3 c9
00000096  01 00 72 3c 46 41 28 e0  8a 89 f1 f0
000000A6  00 05 84 39 10 1f 00 00  83 00 00 00
000000B6  23 13 31 58 00 44 68 15  31 35 07 23
000000C6  05 01 00 01 12 f0 f2 58  0f f1 f9 67
000000D6  05 84 39 10 1d 13 12 20  10 42 66 15
000000E6  f2 f3 40 40 f0 f0 f4 f4  f6 f8 f5 f8
000000F6  f1 f8 f0 f0 f0 f0 f0 f8  f0 f0 f0 f1
00000106  f4 c5 c4 c5 d2 c1
0000010c  d8 c1 f0 f0 6b       e3  d5 c9 c3 c9
0000011c  02 00 70 3c 46 c0 a0 e0  1a 89 f1 f0
0000012c  00 01 23 26 49 7f 01 01  13 00 00 00
0000013c  78 23 15 32 04 07 23 15  12 60 11 05
0000014c  12 f0 f3 22 47 11 f1 f9  67 29 20 48
0000015c  49 7d 15 12 20 10 92 88  24 30 2f f0
0000016c  f6 f5 f1 f0 f0 f0 f0 f0  f0 f0 f3 f1
0000017c  f3 f3 e2 81 95 a3 81 95  84 85 99 40
0000018c  a4 94 85 99 40 c2
```

Wireshark: Display Filter - Profile: share2012_1

Display Filter

- 3-way Handshake
- Delay greater 1 sec
- EE Link up/down
- PING requests and replies
- ctg_fmh7 SENSE
- ctg_snf
- ctg_fmh5 ALLOCATE
- ctg_data GDS12Df
- cics_QA00

Properties

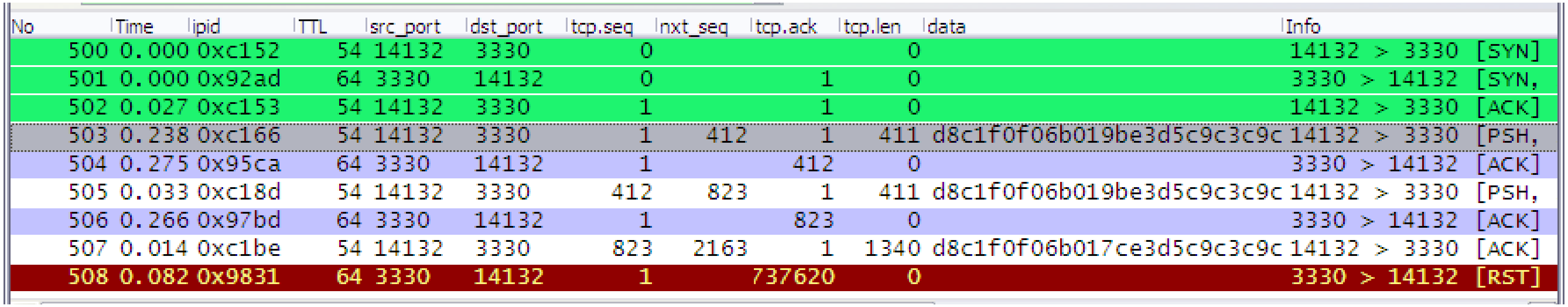
Filter name: cics_QA00

Filter string: data.data[0:4] eq d8c1:f0f0

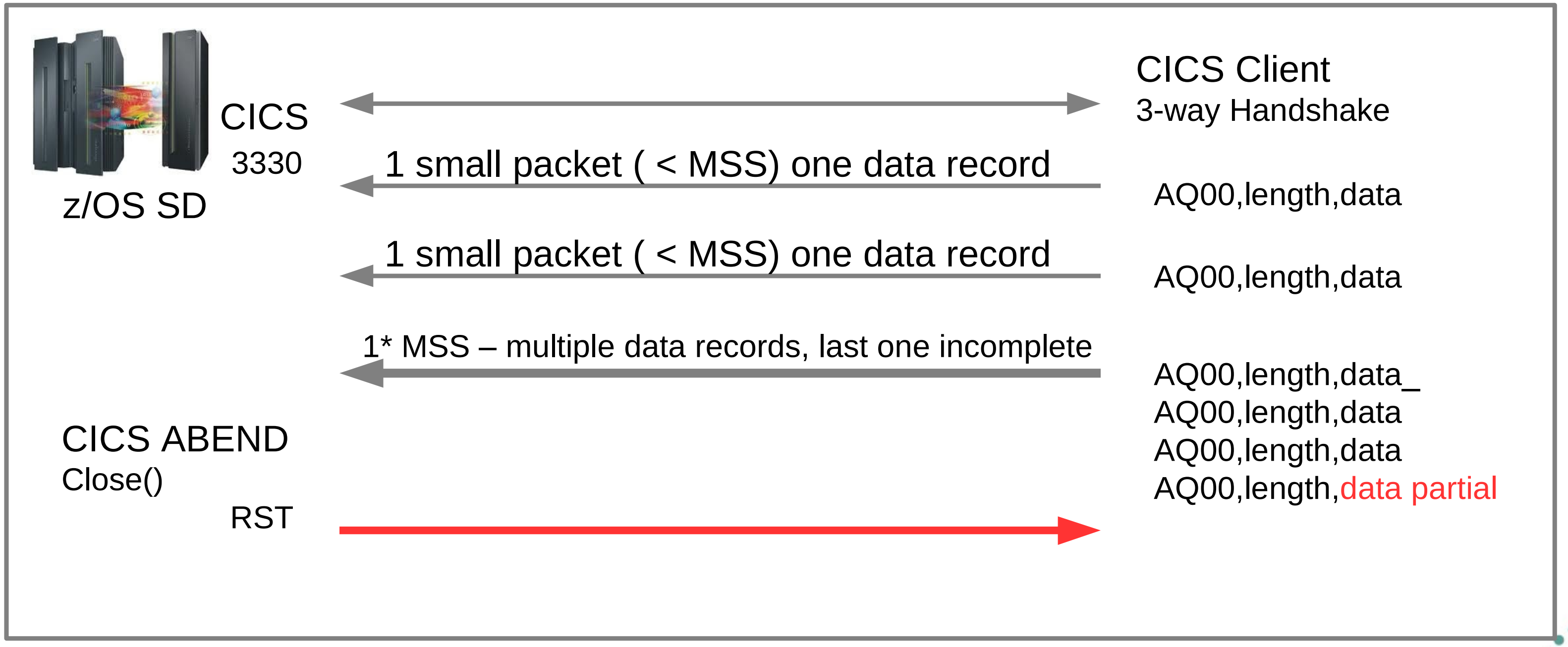
You can create filters to check the data contents:
Here the cics_QA00 filter finds all transactions at the beginning of TCP a segment

Problem 2: CICS ABEND – Flow Chart

Time line



No	Time	ipid	TTL	src_port	dst_port	tcp.seq	nxt_seq	tcp.ack	tcp.len	data	Info
500	0.000	0xc152	54	14132	3330	0			0		14132 > 3330 [SYN]
501	0.000	0x92ad	64	3330	14132	0		1	0		3330 > 14132 [SYN,
502	0.027	0xc153	54	14132	3330	1		1	0		14132 > 3330 [ACK]
503	0.238	0xc166	54	14132	3330	1	412	1	411	d8c1f0f06b019be3d5c9c3c9c	14132 > 3330 [PSH,
504	0.275	0x95ca	64	3330	14132	1		412	0		3330 > 14132 [ACK]
505	0.033	0xc18d	54	14132	3330	412	823	1	411	d8c1f0f06b019be3d5c9c3c9c	14132 > 3330 [PSH,
506	0.266	0x97bd	64	3330	14132	1		823	0		3330 > 14132 [ACK]
507	0.014	0xc1be	54	14132	3330	823	2163	1	1340	d8c1f0f06b017ce3d5c9c3c9c	14132 > 3330 [ACK]
508	0.082	0x9831	64	3330	14132	1		737620	0		3330 > 14132 [RST]



CICS 3330

z/OS SD

CICS Client

3-way Handshake

1 small packet (< MSS) one data record

AQ00,length,data

1 small packet (< MSS) one data record

AQ00,length,data

1* MSS – multiple data records, last one incomplete

AQ00,length,data_
AQ00,length,data
AQ00,length,data
AQ00,length,data partial

CICS ABEND

Close()

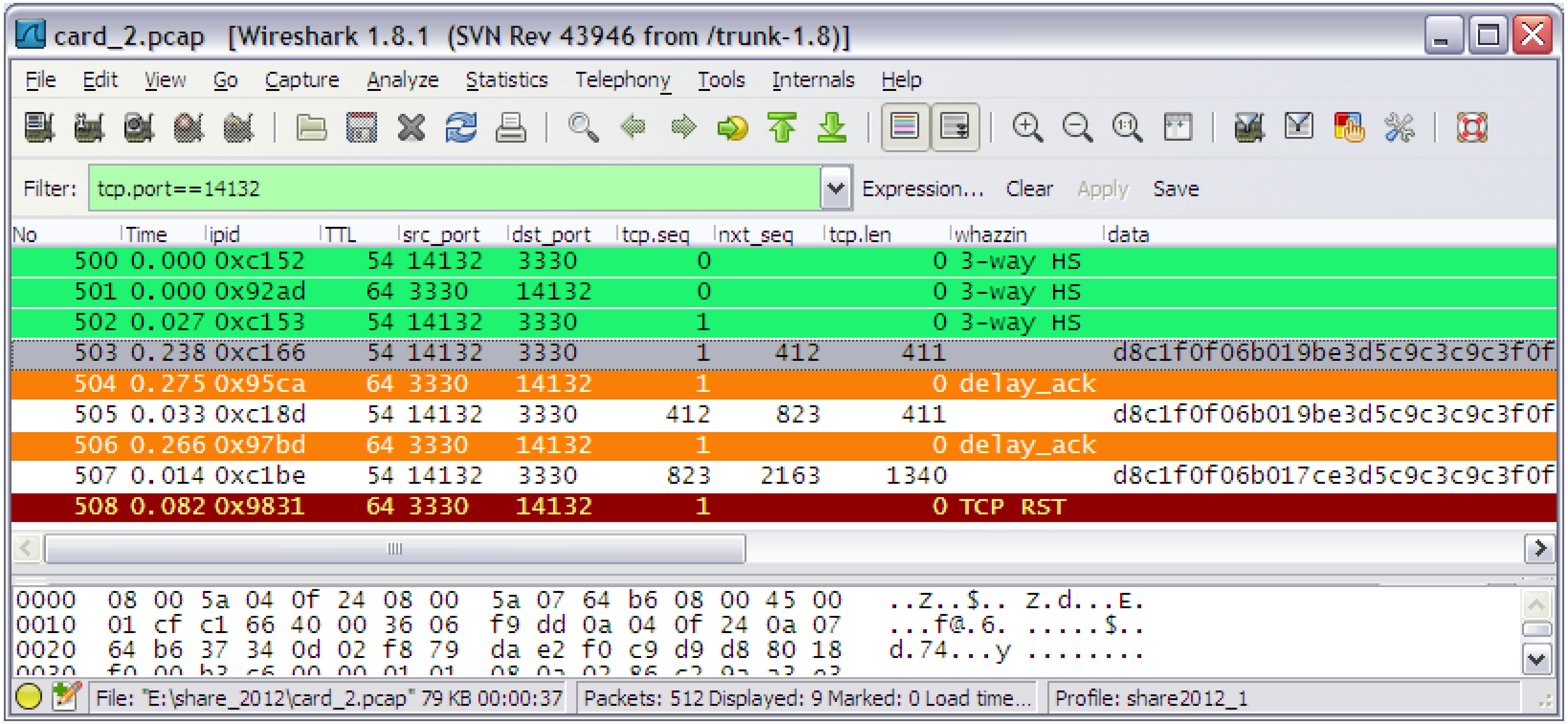
RST

Problem 2: CICS Abend – Problem - Solution

- Understand Problem
  - What is the concern? CICS sessions drop
  - What is the impact? Error messages
  - What is the root cause? Application cannot handle multiple records
- Evaluate possible Solutions
  - Ease of implementation fix the socket application
  - Scope of responsibility avoid multiple data records in a single read()

Problem 2: CICS ABEND NODELAYACK

Delayed acknowledgements



card_2.pcap   [Wireshark 1.8.1 (SVN Rev 43946 from /trunk-1.8)]

Filter: tcp.port==14132

```
No      Time   ipid     TTL src_port dst_port tcp.seq nxt_seq tcp.len  whazzin      data
  500 0.000 0xc152   54 14132   3330       0                   0 3-way HS
  501 0.000 0x92ad   64 3330    14132      0                   0 3-way HS
  502 0.027 0xc153   54 14132   3330       1                   0 3-way HS
  503 0.238 0xc166   54 14132   3330       1     412     411            d8c1f0f06b019be3d5c9c3c9c3f0f
  504 0.275 0x95ca   64 3330    14132      1                   0 delay_ack
  505 0.033 0xc18d   54 14132   3330     412     823     411            d8c1f0f06b019be3d5c9c3c9c3f0f
  506 0.266 0x97bd   64 3330    14132      1                   0 delay_ack
  507 0.014 0xc1be   54 14132   3330     823    2163    1340            d8c1f0f06b017ce3d5c9c3c9c3f0f
  508 0.082 0x9831   64 3330    14132      1                   0 TCP RST
```

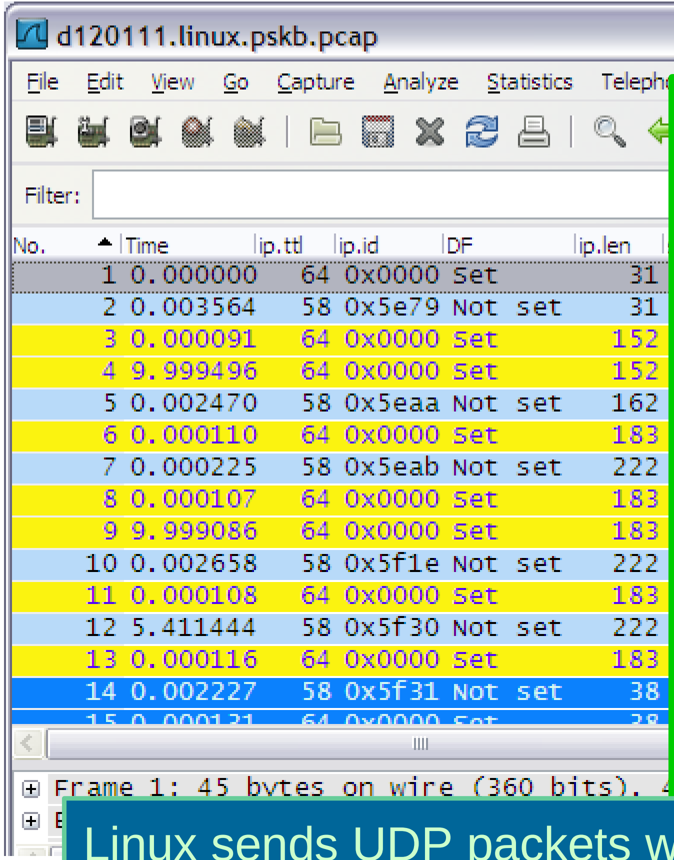
```
0000  08 00 5a 04 0f 24 08 00  5a 07 64 b6 08 00 45 00   ..Z..$.. Z.d...E.
0010  01 cf c1 66 40 00 36 06  f9 dd 0a 04 0f 24 0a 07   ...f@.6. .....$..
0020  64 b6 37 34 0d 02 f8 79  da e2 f0 c9 d9 d8 80 18   d.74...y ........
```

File: "E:\share_2012\card_2.pcap" 79 KB 00:00:37 | Packets: 512 Displayed: 9 Marked: 0 Load time... | Profile: share2012_1

All outbound acknowledgements are delayed by 200 ms
This causes poor performance on the inbund path
Multiple data records are queuing up at the client

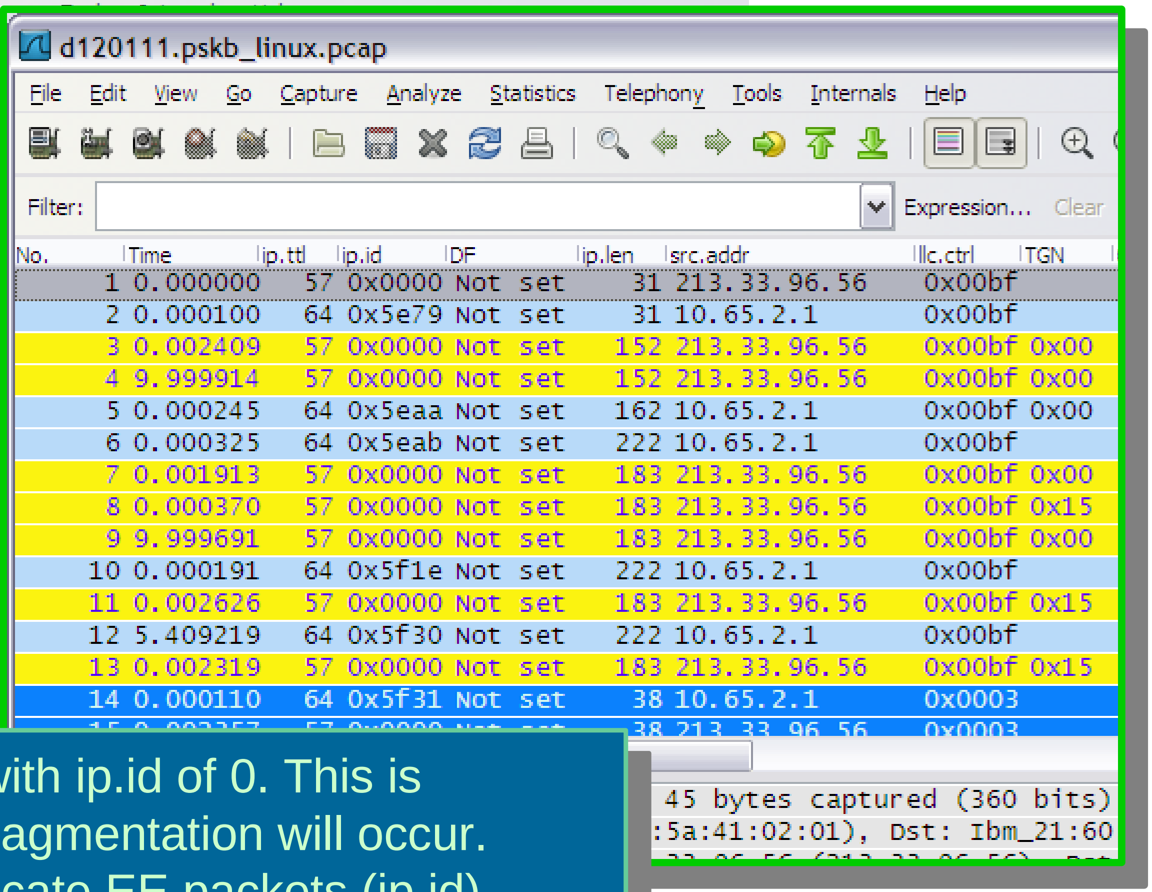
Problem 3: EE Performance Problem

XID takes too long to complete



No.	Time	ip.ttl	ip.id	DF	ip.len
1	0.000000	64	0x0000	Set	31
2	0.003564	58	0x5e79	Not set	31
3	0.000091	64	0x0000	Set	152
4	9.999496	64	0x0000	Set	152
5	0.002470	58	0x5eaa	Not set	162
6	0.000110	64	0x0000	Set	183
7	0.000225	58	0x5eab	Not set	222
8	0.000107	64	0x0000	Set	183
9	9.999086	64	0x0000	Set	183
10	0.002658	58	0x5f1e	Not set	222
11	0.000108	64	0x0000	Set	183
12	5.411444	58	0x5f30	Not set	222
13	0.000116	64	0x0000	Set	183
14	0.002227	58	0x5f31	Not set	38
15	0.000131	64	0x0000	Set	38

```
⊞ Frame 1: 45 bytes on wire (360 bits), 4
⊞ E
```


d120111.pskb_linux.pcap
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression...
Clear

No.	Time	ip.ttl	ip.id	DF	ip.len	src.addr	llc.ctrl	TGN
1	0.000000	57	0x0000	Not set	31	213.33.96.56	0x00bf	
2	0.000100	64	0x5e79	Not set	31	10.65.2.1	0x00bf	
3	0.002409	57	0x0000	Not set	152	213.33.96.56	0x00bf	0x00
4	9.999914	57	0x0000	Not set	152	213.33.96.56	0x00bf	0x00
5	0.000245	64	0x5eaa	Not set	162	10.65.2.1	0x00bf	0x00
6	0.000325	64	0x5eab	Not set	222	10.65.2.1	0x00bf	
7	0.001913	57	0x0000	Not set	183	213.33.96.56	0x00bf	0x00
8	0.000370	57	0x0000	Not set	183	213.33.96.56	0x00bf	0x15
9	9.999691	57	0x0000	Not set	183	213.33.96.56	0x00bf	0x00
10	0.000191	64	0x5f1e	Not set	222	10.65.2.1	0x00bf	
11	0.002626	57	0x0000	Not set	183	213.33.96.56	0x00bf	0x15
12	5.409219	64	0x5f30	Not set	222	10.65.2.1	0x00bf	
13	0.002319	57	0x0000	Not set	183	213.33.96.56	0x00bf	0x15
14	0.000110	64	0x5f31	Not set	38	10.65.2.1	0x0003	
15	0.002357	57	0x0000	Not set	38	213.33.96.56	0x0003	

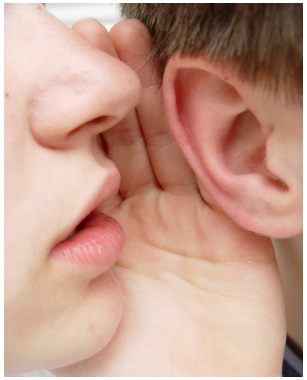
```
45 bytes captured (360 bits)
:5a:41:02:01), Dst: Ibm_21:60
```

Linux sends UDP packets with ip.id of 0. This is possible for as long as no fragmentation will occur. VTAM has a check for duplicate EE packets (ip.id)

Evaluation Forms – IP wizards

We really value your feedback!

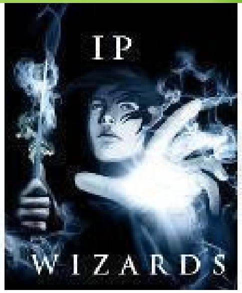
Please take a minute to fill out the evaluation form - leave comments



Register at http://lotus.greenhouse.com

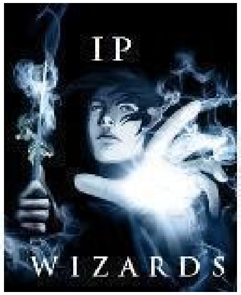
Join the IP wizards community

http://tinyurl.com/ipwizards

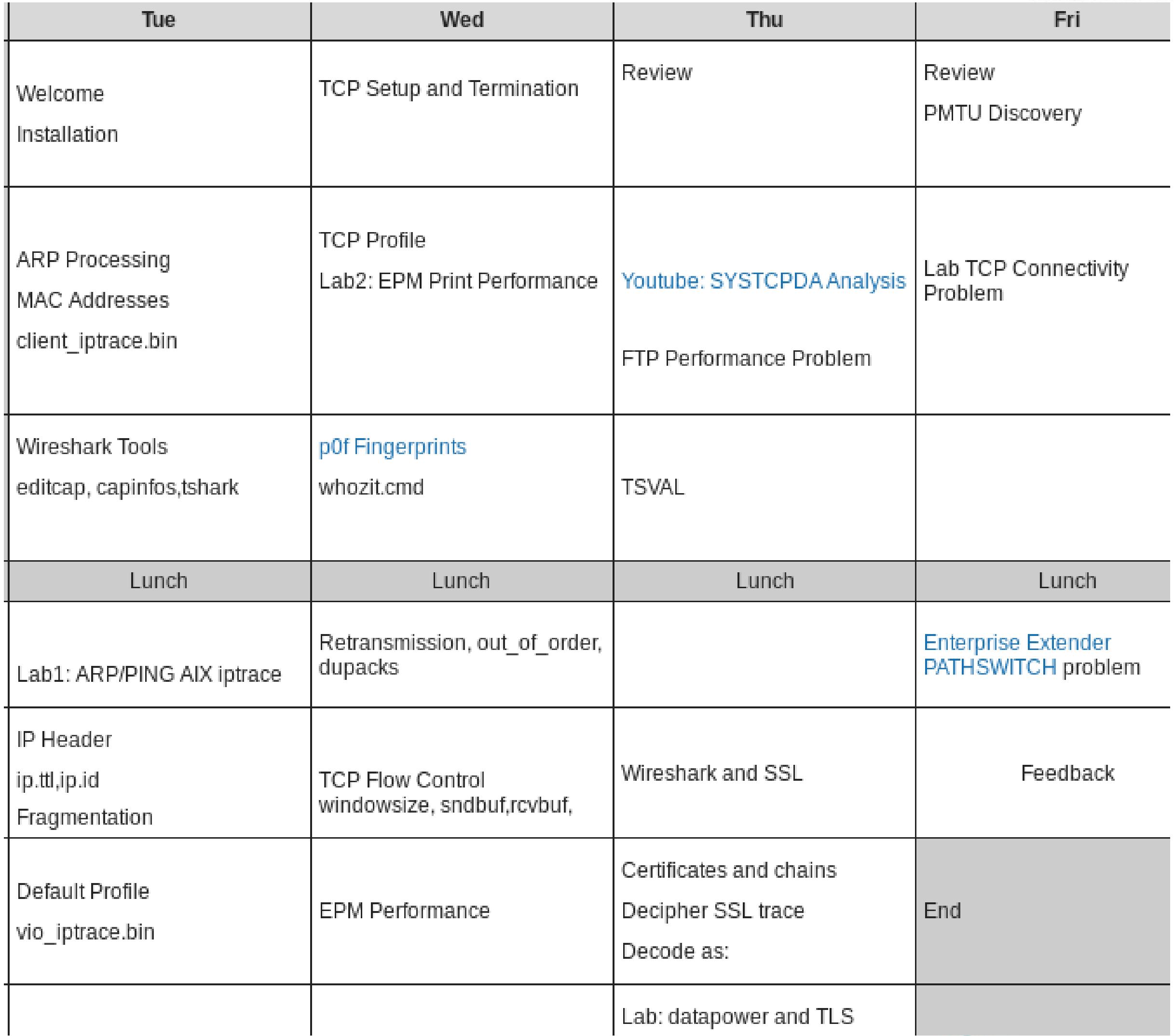


Learn more: Wireshark Bootcamp Agenda

ZOWIE0DE



Turn into a wizard



Tue	Wed	Thu	Fri
Welcome Installation	TCP Setup and Termination	Review	Review PMTU Discovery
ARP Processing MAC Addresses client_iptrace.bin	TCP Profile Lab2: EPM Print Performance	Youtube: SYSTCPDA Analysis FTP Performance Problem	Lab TCP Connectivity Problem
Wireshark Tools editcap, capinfos,tshark	p0f Fingerprints whozit.cmd	TSVAL	
Lunch	Lunch	Lunch	Lunch
Lab1: ARP/PING AIX iptrace	Retransmission, out_of_order, dupacks		Enterprise Extender PATHSWITCH problem
IP Header ip.ttl,ip.id Fragmentation	TCP Flow Control windowsize, sndbuf,rcvbuf,	Wireshark and SSL	Feedback
Default Profile vio_iptrace.bin	EPM Performance	Certificates and chains Decipher SSL trace Decode as:	End
		Lab: datapower and TLS	

http://tinyurl.com/zowie0de

Thank You for your time!

Matthias Burkhard
IBM
mburkhar@ de.ibm.com

Twitter : mreede

IP Wizards

ip.wizards@groups.facebook.com